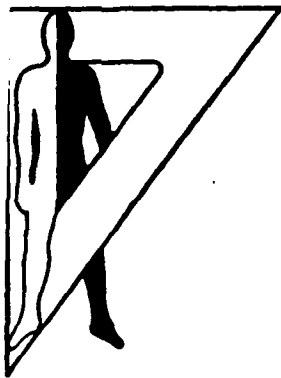


AD-A200 960



AD

Technical Memorandum 7-88

DTIC FILE COPY

HUMAN-MACHINE INTERFACES IN INDUSTRIAL ROBOTICS

H. McIlvaine Parsons
Anne S. Mavor
Essex Corporation

DTIC
ELECTE
NOV 03 1988
S D
CH

September 1988
AMCMS Code 665502.M400000

Approved for public release;
distribution is unlimited.

U.S. ARMY HUMAN ENGINEERING LABORATORY
Aberdeen Proving Ground, Maryland

®AR-BASIC is a registered trademark of American Robot Company.

®UNIX is a registered trademark of Bell Laboratories.

®CIMROC 2 is a registered trademark of CIMCORP.

™CIMPLER is a trademark of CIMCORP.

Destroy this report when no longer needed.
Do not return it to the originator.

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Use of trade names in this report does not constitute an official endorsement or approval of the use of such commercial products.

SECURITY CLASSIFICATION OF THIS PAGE

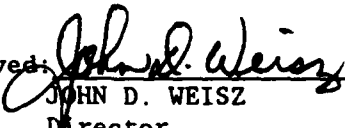
REPORT DOCUMENTATION PAGE				Form Approved OMB No. 0704-0188	
1a. REPORT SECURITY CLASSIFICATION Unclassified			1b. RESTRICTIVE MARKINGS		
2a. SECURITY CLASSIFICATION AUTHORITY			3. DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.		
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE					
4. PERFORMING ORGANIZATION REPORT NUMBER(S)			5. MONITORING ORGANIZATION REPORT NUMBER(S) Technical Memorandum 7-88		
6a. NAME OF PERFORMING ORGANIZATION Essex Corporation		6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION Human Engineering Laboratory		
6c. ADDRESS (City, State, and ZIP Code) 333 North Fairfax Street Alexandria, VA 22314-2691			7b. ADDRESS (City, State, and ZIP Code) Aberdeen Proving Ground Maryland 21005-5001		
8a. NAME OF FUNDING / SPONSORING ORGANIZATION		8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER		
8c. ADDRESS (City, State, and ZIP Code)			10. SOURCE OF FUNDING NUMBERS		
			PROGRAM ELEMENT NO. 6.55.02	PROJECT NO. 1L665502MM40	TASK NO. WORK UNIT ACCESSION NO.
11. TITLE (Include Security Classification) Human-Machine Interfaces in Industrial Robotics					
12. PERSONAL AUTHOR(S) Parsons, H. McIlvaine Mavor, Anne S.					
13a. TYPE OF REPORT Final		13b. TIME COVERED FROM _____ TO _____		14. DATE OF REPORT (Year, Month, Day) 1988, September	
15. PAGE COUNT 118					
16. SUPPLEMENTARY NOTATION					
17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)		
FIELD	GROUP	SUB-GROUP			
23	02		robotics human factors		
05	08		robot controller		
			teach pendant industrial robotics		
19. ABSTRACT (Continue on reverse if necessary and identify by block number)					
<p>This report surveys human-machine interfaces in industrial robotics. Its aim is to create a data base concerning design practices in equipment and software pertaining to human factors engineering, with a summary of programming features and a taxonomy of programming tasks. It results from a study of ten robot manufacturers (eight American, one Swedish, one Japanese), plus a more limited review of another Japanese manufacturer.)</p> <p>The equipment surveyed falls into three categories: teach pendants, cathode-ray tube (CRT)/keyboard terminals, and controller panels; the software consists primarily of the languages and procedures for programming robots on the factory floor as well as in applications programming off-line. The first section reviews aspects relating to programming design from a human factors viewpoint. The second part outlines the component tasks in programming a robot and the uses of these taxonomic descriptions. The Appendix describes in detail the designs in these hardware and software categories for each robot manufacturer and contains illustrations. (KR) ✓</p>					
20. DISTRIBUTION / AVAILABILITY OF ABSTRACT <input type="checkbox"/> UNCLASSIFIED/UNLIMITED <input checked="" type="checkbox"/> SAME AS RPT. <input type="checkbox"/> DTIC USERS			21. ABSTRACT SECURITY CLASSIFICATION Unclassified		
22a. NAME OF RESPONSIBLE INDIVIDUAL Doris S. Eanes			22b. TELEPHONE (Include Area Code) 301-278-4478		22c. OFFICE SYMBOL SLCHE-SS-IR

HUMAN-MACHINE INTERFACES IN INDUSTRIAL ROBOTICS

H. McIlvaine Parsons
Anne S. Mavor

Essex Corporation
333 North Fairfax Street
Alexandria, VA 22314-2691

September 1988

Approved: 

JOHN D. WEISZ

Director

Human Engineering Laboratory

Approved for public release;
distribution is unlimited.

U.S. ARMY HUMAN ENGINEERING LABORATORY
Aberdeen Proving Ground, Maryland 21005-5001

FOREWORD

As indicated in this survey report, a number of robot control pendants (or other operator interfaces) use light-emitting diodes (LEDs) as indicators. The range of colors that can be produced cheaply with commercial LEDs is limited and the least expensive LEDs produce red light. As a result, some of the LED colors used by industrial robot manufacturers are different from those recommended by MIL-STD-1472 (Department of Defense, 1981); in some instances red LEDs are being used to indicate normal operating ranges or normal operating parameters, whereas MIL-STD-1472 (Department of Defense, 1981) recommends that the color red be reserved for warning indicators. This potential conflict between commercial practice and good human engineering practice was encountered recently by the Robotic Industries Association (RIA) 15.02 Committee on Human Interface, which is preparing a document titled Human Engineering Design Criteria for Hand-Held Robot Control Pendants. The 15.02 Committee accepted the color-coding recommendations found in MIL-STD-1472 (Department of Defense, 1981) for use on a voluntary consent basis, except that red can be used to indicate normal status. However, flashing red was recommended only for use as a warning indicator.

David C. Hodge
Member, RIA 15.02



Accession For	
NTIS GRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

ACKNOWLEDGMENTS

We wish to express our appreciation to Dr. David C. Hodge of the U.S. Army Human Engineering Laboratory for his support in carrying out this contract and to Dr. T. Michael Knasel, Vice President and General Manager, Automated Systems Division, The MK-Ferguson Company, for helpful consultation. In addition, we thank those representatives of robot manufacturers cited in the report and others who have not been identified for providing useful information. All the companies were given the opportunity to review their sections and suggest changes.

CONTENTS

EXECUTIVE SUMMARY	3
PART ONE: SUMMARY OF SURVEY DATA ON PROGRAMMING FEATURES	5
Introduction	5
Teach Pendants	7
Other Design Aspects of Teach Pendants	11
Computer Terminal	12
Controller Panel	13
Allocation of Programming Tasks	13
Software	14
Documentation	15
Human Factors Interests	16
PART TWO: GENERIC FUNCTIONS AND TASK ANALYSIS OF ROBOT PROGRAMMING	17
Section A: Outline of a Suggested Task Taxonomy	17
Preliminary Operations	17
Specification of Each Robot Motion/Action	18
Sequencing Robot Motions/Actions	25
Conversion of Robot Motions/Actions and Their Sequencing Into a Higher-Order Language (Coding): Writing and Applications Program	26
Program Modification	26
Program Administration	27
Section B: Uses of a Performance-Oriented Analysis	28
Design Checklist	28
Comparisons	28
Programmer Effectiveness	28
Design Innovations	29
Neglected Aspects	30
New Notions	32
Desirable Investigations	32
Skill Requirements	37
Manuals and Handbooks	37
CONCLUSIONS	38
REFERENCES	39
APPENDIX	
Survey of Eleven Robot Manufacturers' Equipment and Software Including a Description of Interface Features	41
1. Adept Technology, Inc.	43
2. American Cimflex Corporation	48
3. ASEA Robotics, Inc.	56
4. Cincinnati Milacron	63

5.	CIMCORP (formerly GCA Corporation)	75
6.	GMF Robotics Corporation	82
7.	International Business Machines Corporation	90
8.	Schrader Bellows Division of Parker Hannifin Corporation	94
9.	Toshiba Corporation, Tokyo, Japan, and Houston, Texas	98
10.	Unimation, Inc. (Westinghouse)	103
11.	Hitachi, Ltd., Tokyo, Japan	111

TABLES

1.	Motion Control of Eight Manufacturers	8
2.	Velocity Control of Eight Manufacturers	10
3.	Particular Design Innovations That (May) Have Helped Programmer Performance	29
4.	Instances of Error Management	31
5.	Design Aspects That May Need More Investigation	33
6.	Configurations	35
7.	Menu Systems (Seven Robot Manufacturers)	36

EXECUTIVE SUMMARY

This report surveys human-machine interfaces in industrial robotics. Its aim is to create a data base concerning design practices in equipment and software pertaining to human factors engineering (ergonomics), with a summary of programming features and a taxonomy of programming tasks. It results from a study of ten robot manufacturers (eight American, one Swedish, one Japanese), plus a more limited review of another Japanese manufacturer. The study, which began in July 1985, was funded by the U.S. Army Human Engineering Laboratory under a Phase I contract in the Small Business Innovative Research (SBIR) program.

The equipment surveyed falls into three categories: teach pendants, cathode-ray tube (CRT)/keyboard terminals, and controller panels; the software consists primarily of the languages and procedures for programming robots on the factory floor as well as in applications programming off-line. The Appendix describes in detail the designs in these hardware and software categories for each robot manufacturer and contains illustrations. The first section reviews aspects relating to programming design from a human factors viewpoint. This report does not systematically describe human factors studies or analyses that the robot manufacturers have performed with respect to their products; there have been very few under that label, though it should not be assumed that ease of use has been entirely disregarded. Programming (teaching) and operation are the activities emphasized, to the neglect of maintenance, sensor, and safety procedures and devices (despite their importance). Also omitted are significant human factors areas such as task allocation between robot and human, factory training, and test and evaluation (e.g., during robot installation). Design considerations were what seemed most applicable to the Army's concerns about soldier-machine interfaces in field applications of robotics. The second part of this report outlines the component tasks in programming a robot and the uses of these taxonomic descriptions. This outline may help the reader understand many of the design features in the first part and thus might be read first. Or it can be considered independently as an attempt to fill a gap in the literature about robot programming.

To begin it may be helpful to discuss how the four hardware and software categories relate to each other. Teach pendants are portable control and display panels that a technician or engineer takes to the vicinity of the robot's end effector if it becomes necessary (as it often does) to be visually close while directing the robot's motions in programming it. (In another manual programming method, a craftsman moves the robot directly or by a surrogate robot. Such is the technique for programming with controls on the robot's or surrogate's arm/hand. A relatively simple assembly robot may also be guided manually if a CRT/keyboard terminal is nearby. This type of manual teaching is not stressed in this report.) A teach pendant (TP) is connected to the robot controller for outputs and inputs. Various switches on that unit's panel control overall robot operations, including starting and stopping it during production. The controller houses a microprocessor and the applications program that is taught or otherwise programmed and that subsequently guides the production run; that program and its language constitute the software of interest in this study. Like the teach pendant,

the CRT/keyboard terminal is connected to the controller for inputs and outputs and also to the applications program. The terminal may handle more nonmotion programming than the pendant, which is the primary device for teaching the robot manipulator what movements to make. In many robot installations, notably assembly, only a small proportion of an applications program consists of robot motions. A major design issue is the allocation of programming tasks (including duplication) between pendant and terminal, which can also provide status information during the subsequent production run that the applications program controls. Much of that program can be created on an off-line computer, even including some of the robot's movements through CAD (computer-aided design) simulation. The operating system software is also off-line.

A number of other U.S. robot manufacturers were queried in this study in addition to those in the equipment and software descriptions discussed later, which are based on operation and programming manuals (not all as comprehensive and coherent as we would wish) and visits to five companies. The other manufacturers were omitted because they used only the direct method of manual teaching, they primarily produced vision systems or educational robots, or they failed or declined to provide sufficient descriptive information. Undoubtedly some manufacturers that should have been included were never queried.

The data base has some omissions, such as dimensional data about teach pendants--weight, area, display characters, push-button size, and spacing between buttons. Such data are usually omitted from manuals and brochures. A study of 10 pendants by Levosinski (1984) indicated some pendants were "too large to handle with one hand," and "many pendants weigh 8 to 15 pounds," but generally pendants are much lighter, even weighing as little as 2 pounds. A pendant must be ruggedly constructed since it is likely to be dropped from 1 or 2 meters onto a concrete floor. No two pendants are alike. Smola (1986, p. 18) noted that "Originally, robot manufacturers were on their own when it came to teach pendant design and manufacturing. Today, there are off-the-shelf, handheld terminal packages that can be tailored to include features unique to a given manufacturer." In comparison, controller panels are relatively simple interfaces but still have human engineering requirements. For example, the visibility of status indicators on the controller panel has seldom been specified. Shulman and Olex (1985) found one panel on which such indicators could not be easily seen at a sufficient distance. Anthropometric and perceptual aspects of the CRT/keyboard terminal have seldom been noted in industrial robotics, although these can induce postural and visual fatigue, as human factors studies on video display terminals (VDTs) have demonstrated (Galitz, 1984).

HUMAN-MACHINE INTERFACES IN INDUSTRIAL ROBOTICS

PART ONE: SUMMARY OF SURVEY DATA ON PROGRAMMING FEATURES

INTRODUCTION

Human-machine interfaces in industrial robotics have seldom been examined, at least in any comprehensive fashion. Why? Perhaps it is because robots exemplify automation, and as autonomous devices their use would not seem to require human participation. But it does, as pointed out by Parsons and Kearsley (1982), Noro and Okada (1983), and Salvendy (1983). On the factory floor humans operate robots, maintain them, and program them. In addition, some industrial tasks in some situations are shared between robots and workers--for example, inspection, assembly, dealing with inputs to robotic handling, and outputs. Another human factors interest is safety.

This section describes the most demanding and interesting of the human tasks--robot programming. Funded by the U.S. Army Human Engineering Laboratory at Aberdeen Proving Ground, Maryland, a survey was conducted among ten major robot manufacturers--eight American, one Swedish, and one Japanese--during 1985 and 1986. (Some information was obtained from another Japanese manufacturer, and other American companies were queried but for one reason or another did not provide useful information.) The ten sources were Adept Technology, American Cimflex Corporation, ASEA, Cincinnati Milacron, GCA Corporation (now CIMCORP), General Motors Fanuc (GMF), International Business Machines (IBM), Schrader Bellows Division of Parker Hannifin, Toshiba, and Unimation/Westinghouse. The individual companies are not identified further in this report, so that an overall picture rather than a critique of particular robot manufacturers can be presented. The Appendix contains detailed data about each company from interviews and documentation (such as manuals).

Three interrelated hardware interfaces and one software interface are involved in programming an industrial robot, all described in this report. The three hardware interfaces are (1) a teach pendant (also called a teach box or programming unit), (2) a computer terminal, and (3) a controller panel. The software interface is the applications program that has to be created to make the robot perform. The teach pendant is a hand-held control/display panel that is carried inside the robot's work envelope. The terminal is operated outside an area that is usually fenced off, although a portable terminal may be carried inside. The pendant and the terminal are connected to a microcomputer inside the controller and outside the controller; its panel is used primarily in robot operation. The application software is programmed by means of the pendant and terminal as well as elsewhere by a terminal, computer-aided design (CAD) simulation, or downloading from another computer.

Programming a robot is a relatively complex process that has become more so in recent years with sophisticated algorithms for robot motion, interaction with other machines, reliance on various sensors, and multiplicity of robot movements. The movement commands have special interest because a programmer

must generate them in a manner peculiar to robotics. The programmer customarily uses the robot itself to create commands that will subsequently move it when the program is executed, though help may come from sensors, prior programs, or CAD simulation.

For a robot to move the end effector or tool (gripper, weld gun, etc.) to a point or to a series of points where the tool will do its work then the program must have acquired (learned) those locations precisely in three-dimensional coordinates. The robot must also have acquired the angular orientation that the tool should have in working on the target object, perhaps also in three dimensions. The most effective way to acquire these dimensional data is to move the robot's tool center point to each target point and also orient the tool appropriately. The origin and robot's home position and orientation in a reference frame of coordinates are known after the robot has been calibrated at the controller panel. The new position in that frame can be derived trigonometrically from the changes that occur in the robot's joint angles, which are computer-calculated from the joint servos. The new orientation of the tool can also be derived from joint angles.

There are two ways to move a robot to a point and orient it. One is to push or pull it. This suffices for small assembly robots or for spray-painting robots (for which a light surrogate robot may be substituted), but not for large robots with heavy tools. These are moved by the same joint motors that move them when a program is executed. The joint motors are activated by controls on the teach pendant that the programmer brings very close to the tool center point and target point. To align these with precision is essential but difficult if not impossible to achieve. That is why the pendant must be operated inside the robot's work envelope. It must be small and light enough to carry around.

When the tool center point has been precisely aligned with the target point and the tool has been properly oriented, the programmer presses a pendant RECORD button and the position and orientation are programmed. The process resembles teleoperation, except that the programmer relies on direct human vision rather than a television camera and monitor and postpones execution of the robot's action rather than undertakes it then and there.

The process is also more elaborate than this simplified description suggests. The programmer must select the velocities (and their variations) at which the joint motors will move the robot. He must move the robot along three rectangular (Cartesian) axes in space (in one of two frames of reference--world or tool) and orient it in yaw, pitch, and roll (with the position maintained at the alignment point). As an alternative, the programmer might move each joint independently. In the past this was the sole method of moving a robot to a point and giving it an orientation; it is still used by one of the manufacturers surveyed as its only method. It is still useful for maintenance, for orientation, and for some precision positioning. The robot's required path usually calls for a sequence of points to be programmed, with stops, pauses, or short cuts between them. Resumption from a pause may be conditional on a signal from another machine (e.g., a conveyor) or a sensor. Programming these aspects can be accomplished at the terminal or off-line instead of at the pendant. So can selecting the kind of trajectory (straight-line or curved) and movement speeds during execution.

Industrial robots have up to six joints ("degrees of freedom") that are revolute (rotational) or prismatic (translational), plus end effector activation (e.g., open or close a gripper); small assembly robots may have only two to four joints. To position the tool center point, the programmer moves it along the X-, Y-, and Z-axes in a world coordinate frame of reference (or in a tool coordinate system) and the wrist joints are actuated for tool orientation in pitch, yaw, and roll. In short, the programmer must accomplish a variety of robot movements with a teach pendant and edit these, perhaps after replay, with deletions, revisions, and insertions.

TEACH PENDANTS

As noted in the Introduction of Part One, this survey did not systematically consider teach pendant size or weight, though these are important considerations. All pendants appear to be more or less rectangular, some with a longer horizontal axis, some with a longer vertical axis, and some with a T or an inverted L shape. Methods of grasping vary. One has a slot through which fingers are inserted. One has a cord to fit around the hand to prevent dropping. At least one manufacturer explored a number of designs though apparently without systematic testing. All, except the pendant that represented early simpler programming, have some kind of liquid crystal display (LCD) in addition to light-emitting diodes (LEDs) associated with buttons or that function as status indicators. The display's capacities vary from one line of characters to eight and the number of characters per line from 3 to 40. Two displays have windows or sections, and two incorporate scrolling. Contents also vary greatly, with no two displays having just the same items. Categories include program identification; operations mode; coordinate system; the name, type, and coordinates of a point; commands and arguments that have been inserted; error messages; menu options; and prompts.

Control Features

All the teach pendants surveyed have many push buttons (keys) or selector switches on their surfaces and have several selector switches on a side. The totals (with the number of commands, arguments, modes, or other functions these can input in parentheses) are 21 (89), 23 (45), 25 (25), 25 (40), 28 (43), 32 (32), 38 (32), 42 (47), and 46 (86). The totals for control elements in two cases include joysticks and in four instances current or planned soft keys for responding to menus shown on the LCD; another manufacturer has a menu system in which responses to displayed options can be made through hard keys. The command totals do not include those available through soft keys. Most manufacturers incorporate numeral keys (buttons), some in a distinct keypad, some accessible by shift keys. The totals of control elements include an emergency stop button for most and a deadman switch for some. In instances where the number of commands exceeds the number of control elements, one or more shift keys invest two or more commands in a single button. The same multiplicity of functions occurs of course with the soft buttons, though not reflected in the totals. In fact, a menu system with soft buttons is an alternative to a large number of single-function hard keys or a smaller number of multiple-function hard keys. Menus require a display area affecting the LCD size.

Different functions for buttons are enabled by a designated shift key or several shift keys and, in a few instances, by repeated pressing of a button or by a sequence of operations in the program with a display of the alternative function. Multiple use particularly characterizes buttons or sticks for making the robot move in programming it. As shown in Table 1, selector switches are used to assign to buttons or a joystick, robot motion in world coordinates, tool coordinates, or by individual joints. The buttons or stick movements are allocated to particular axes within coordinate systems or to particular joints, and the two directions of robot movement depend on which button is pressed, on the use of selection buttons, or on the direction of stick movement.

Table 1

Motion Control of Eight Manufacturers

-
- (1) Six rocker switches in a column, one for each axis or joint
 Upper three for world coordinates or arm joints
 Lower three for tool coordinates or wrist joints
 Left rocker side for one direction, right for other
 Four selection buttons for world coordinates, tool coordinates, joints, or free mode (push/pull)
 - (2) Twelve paired buttons in two sets of six each (4 x 3 array)
 Left set moves arm by coordinates or joints
 Right set moves wrist by coordinates or joints
 Within sets: different row for each axis or joint
 Within sets: different column for each direction
 Shift key selects between coordinates and joints
 - (3) Twelve paired buttons in two sets of six each (4 x 3 array)
 Left set moves arm by coordinates or joints
 Right set moves wrist by coordinates or joints
 Within sets: different row for each axis or joint
 Within sets: different column for each direction
 Three buttons select between base coordinates or tool coordinates or joints
 - (4) Twelve paired buttons in two sets of six each (4 x 3 array)
 Left set for positioning, right for orientation
 Within sets: different row for each axis or joint
 Within sets: different column for each direction
 Two buttons (at controller panel) select between teach (coordinates) mode and manual (joints) mode

(continued on next page)

Table 1 (continued)

- (5) Twelve buttons in two columns, six each
 Different row for each axis or joint
 Upper three for arm, lower three for wrist
 Left column for one direction, right for other
 Two buttons select between world coordinates and joints
- (6) Six buttons in a column (plus two for gripper); four for
 4-degrees-of-freedom (DOF) robot
 Upper three buttons enable robot movement in world coordinates
 or arm joints
 Lower three buttons enable robot movement in tool coordinates
 or wrist joints
 Different button for each axis or joint
 Two elongated buttons (also move robot and set speed) determine
 the direction robot moves
 Single button (pressed in succession) selects between world
 coordinate frame, tool coordinate frame, individual joint
 control, and free mode (push/pull)
- (7) Joystick is moved side-to-side, forward-and-back, or rotated
 Side-to-side moves robot in Y-axis in world or tool coordinates
 Side-to-side orients wrist to tool center point around tool's
 Z-axis
 Forward-and-back moves robot in X-axis in world or tool
 coordinates
 Forward-and-back orients wrist to tool center point around tool's
 Y-axis
 Rotation moves robot in Z-axis in world or tool coordinates
 Rotation orients wrist to tool center point around tool's X-axis
 Direction of stick movement or rotation determines the direction
 robot moves
 Toggle switch selects between coordinate system mode or tool
 center point orientation mode
 Two buttons select between world or tool coordinate frames
- (8) Joystick is moved side-to-side, forward-and-back, or rotated
 Side-to-side moves robot in Y-axis in world or tool coordinates
 Side-to-side orients wrist to tool center point
 Side-to-side rotates robot's wrist joint
 Forward-and-back moves robot in X-axis in world or tool
 coordinates
 Forward-and-back orients wrist to tool center point
 Forward-and-back rotates robot's shoulder joint
 Rotation moves robot in Z-axis in world or tool coordinates
 Rotation rotates robot's elbow
 Direction of stick movement or rotation determines the
 direction robot moves
 Selector switch selects between world coordinates (frame/arm joints)
 or tool coordinates (frame/wrist joints)
 Button selects between coordinate frame control or joint control

In some cases two or more movement buttons may be pressed at the same time or a stick may be moved along more than one axis, so the robot then moves along two or more axes, or two or more joints rotate or translate at the same time. Also, the trajectory the programmer uses to position the robot is not the same as the trajectory the robot takes after it is programmed. The trajectory is calculated by a computer program to be either a straight line from point to point or a "joint-interpolated" or "joint-coordinated" motion, which is a curved line with joint movements ending at the same time. Curves are usually programmed more by teaching small straight-line segments.

Similarly, the velocities the programmer uses in teaching the robot are not the same as those the robot assumes when the program is executed. Execution speeds are, in most cases, programmed at a terminal and exceed programming (and playback) speeds, which are slower to minimize hazards to the programmer operating the teach pendant within the robot's operating zone. Table 2 shows the variety of ways in which robot manufacturers have designed speed control for programming.

Table 2

Velocity Control of Eight Manufacturers

-
- (1) Three buttons: low, medium, high
 - (2) Three-position switch: normal, slow, high
Button slows each of these
 - (3) Two buttons increase or decrease displayed speed by 5 percent with each press
 - (4) Three buttons: high (50 percent of maximum), medium (10 percent), low (1 percent)
 - (5) Button increases speed
Button lists available speeds for selection or change
Keyboard input sets speed
 - (6) Declination or extent of rotation of joystick determines speed logarithmically
 - (7) Switch settings from 0 to 10 establish speed
 - (8) Declination or extent of rotation of joystick determines speed within each setting
 - (9) Location of press on two elongated buttons (also for movement actuation and direction) determines speed, with graphics guidance
 - (10) Toggle switch alters speed ranges within buttons
-

OTHER DESIGN ASPECTS OF TEACH PENDANTS

In addition to those discussed previously, these hand-held control pendants have a number of design aspects of particular human factors interest.

Grouping

In general, buttons with related functions are placed in proximity to each other, by row or column, or related functions are vested in the same multiple-use button. In some pendants, groups of functionally related buttons are spatially separated from each other; in others, all buttons are arranged in the same matrix with no separation between groups.

Coding

Button grouping with or without spatial separation is a form of position coding. In other position coding, switches in a few instances are placed on the side rather than the surface of the pendant; in addition they differ in shape from the buttons. Other occasional shape differences are rocker and toggle switches. Emergency stop buttons are usually larger and may be placed outside the button matrix. They also are customarily color-coded red. Color-coding of buttons is used extensively by several manufacturers (often with grouping), to a limited extent by some, or not at all by others. Button labels on one company's pendant that have numerous functions per button are color-coded to match the colors of several shift keys.

Labeling

Labels on buttons are generally words, abbreviations, or letters and numerals. Symbols are infrequent except for simple ones like plus and minus signs and arrows, but one manufacturer, whose robots are distributed in many countries, makes extensive use of symbols that might be regarded as the robotic equivalents of international road signs. Abbreviations occur in some pendants when the entire word has more than five or six letters, in some when it has more than four, and in one when it has more than three. (The shorter the abbreviation, the greater the chance for confusion.)

Feedback

Tactile feedback of control activation occurs with mechanical buttons and selector switches, and with membrane buttons that give a tactile signal when pressed; all these types are found on the pendants surveyed. One manufacturer's pendant beeps when a key is pressed, and another's beeps several times and the display blinks when the programmer presses the RECORD button. In many instances an LED next to the button lights up. Some button commands or computer responses to a command appear on some pendant LCDs, depending in part on its size and capacity. In another kind of feedback, some menu selections by soft keys advance the menu to a further level.

Errors

Several pendant displays show errors. In one case errors are displayed by a code that can be interpreted by examining a printed aid. Other pendants do not show errors because of the limited space on the display. However, on one pendant errors are indicated by the menu system. One manufacturer has an alarm LED, and on another manufacturer's pendant, the LEDs flash. Errors, however, are more likely to be indicated at the computer terminal or controller panel.

Special Control Devices

As already indicated, two of the manufacturers surveyed have placed joysticks on their pendants in place of buttons to move the robot in programming it. However, very recently one of these pendants has been redesigned, replacing the stick with push buttons and many push-button functions with a menu system, including soft keys. Another manufacturer, in an early prototype, mounted three joysticks in a box attached to a handgrip (from a helicopter) but abandoned this design when users said it was too awkward. Still another manufacturer briefly considered using a joystick instead of buttons in its current pendant. In another pendant, two unusually shaped button-type controls both initiate robot motion and establish its velocity, which depend on where the programmer presses the control, with graphics indicating increasing velocity; one control moves the robot's joints in one direction, the other in the opposite direction. Since the robot moves only while one of these controls is being pressed, it functions as a deadman switch. Some other pendants have deadman switches, enabling pendant movement only when holding and pressing the pendant or by resting the palm of your hand on it while operating a joystick; removing your hand cuts power to the robot's joint motors.

Only one pendant has buttons with the letters of the alphabet, combined with another function on each button. These alphabet buttons can be used to enter commands in a menu system instead of soft buttons, which are absent. This manufacturer does not make use of a fixed computer terminal with a keyboard stationed outside the robot's working area, though it does have a portable one as an option.

COMPUTER TERMINAL

The portable terminal just mentioned is not hand-held but placed on a stand. It has 15 dedicated command buttons with the same functions as the pendant and a letter keyboard (in alphabetical rather than QWERTY format) that is not shared with other functions; it differs by lacking any control elements for moving the robot and by having a larger CRT display showing data being entered and data already entered and/or programmed. Another manufacturer among those surveyed has a portable terminal only, with an LCD and letter and numeric keys as well as instruction keys. This terminal also has control elements for programming the robot's movements. Two other robot manufacturers in the survey lack either a fixed or a portable computer terminal connected to the robot controller. One of these manufacturers produces relatively simple

robots similar to those typical a decade ago. The other achieves versatility through a menu system managed at the teach pendant, whose display can show 2 lines of 40 characters each.

Six of the companies have CRT-equipped computer terminals placed outside the robot's operating area. Four of these use menu systems in addition to keyboard commands. A fifth presently uses only commands and instructions with references to a user's manual. Eight of this terminal's letter keys can issue special commands when shifted by a control key; they are designated by single letters. The sixth company's terminal is similar. The terminals that can use menu systems have soft keys varying in number between companies. On some of the six terminal keyboards some of the keys are color-coded, and at least one company color-codes its display. The CRT screens vary in their formatting; one has windows and another is sectioned.

CONTROLLER PANEL

Variation among the robot manufacturers is also notable in the designs of their controller panels, whose primary role, as noted earlier, is robot operation rather than programming. Panels have various controls, with associated indicators and procedures. These controls are used

- for selecting operational modes,
- for calibrating the robot,
- for turning power on and off to various system components, and
- for starting, interrupting, restarting, and stopping program execution including emergency stops, brake overrides, and end-of-cycle stops and cycle starts.

The computer terminal is usually close to the controller panel and also can monitor program execution. The robot operator is usually a different person from the robot programmer but, since some operation is necessary before programming can be undertaken and program revision may be required in the course of operation, the two roles may at times be combined.

ALLOCATION OF PROGRAMMING TASKS

As programming a robot has become more complex, requiring more control elements and greater display size, robot manufacturers have been facing a dilemma. What programming tasks should be allocated to the teach pendant, what to the computer terminal? Since it has seemed difficult to assign all tasks to either one or the other, the programmer must work with each, moving back and forth between them (and the controller panel), and in and out of the robot's work envelope. Often, for safety, that envelope is enclosed by a barrier with a gate. This dilemma has been resolved in various ways. One is to increase the number of control elements on (and weight of) the pendant. Rather than enlarge the pendant, another is to give its elements multiple uses

or substitute a joystick for buttons. A third is to use soft keys and a menu system or include letter keys for some of the control elements. As control elements increase or a menu system is instituted, the pendant's display capacity as well as its size must also increase, further adding to the pendant's weight. As an alternative, the computer terminal with its keyboard and large display is made portable so it can be brought inside the robot's work envelope and placed on a stand near where the programmer is working with the pendant. Different manufacturers have chosen different alternatives.

Miniaturizing the pendant's control and display elements might also be a possibility but then the programmer might make errors in manipulation and discrimination. (Current error rates are unknown. Sizes of control and display elements and widths of separations between elements are not specified in manufacturers' documentation.) Conceivably the programmer might align and orient a tool center point with its target point from outside the robot's work envelope by closed-circuit television (CCTV), with a camera on the robot's wrist and a monitor at the computer terminal. But this solution apparently has not been advanced, much less tested, though robot-mounted cameras are used extensively for machine vision. In some situations no dilemma is apparent. For example, in small-part assembly with a small robot, often with only a few degrees of freedom, the programmer can work at a computer terminal very near the robot and directly view the point-to-point alignment. The programmer does not need a pendant since the robot can be moved by hand (or the terminal may have keys for moving it).

SOFTWARE

An applications program includes commands and their arguments (modifiers), other instructions, menus, logical and arithmetic operations, error detection and correction, and editing capabilities. It also includes user-aiding components such as prompts, command lists, error feedback, and responses to help requests.

The applications software for nine companies using such software reflects a remarkable variation in categorization of commands. For example, one manufacturer lists 12 categories with 92 commands and functions including file, editing, manipulating scalar variables, algebraic functions, position definition commands, position definition functions, setting robot parameters, motion commands, program logic, input/output, execution and debugging, and system commands. Another's documentation identifies nine categories of instructions and functions: editing, positioning, control, automatic, manual system control, adaptive control, and others for arc welding, links to superior computers, and a vision system. An analysis of a third company's documentation suggests 65 or more commands in four categories: teaching, function, examine, and menu. For a fourth, categories include motion control, assignment, edit, sequential control, and communication commands. A fifth manufacturer's software has 176 commands and instructions in 19 categories. (The category labels in all of these summaries are taken from company documentation.)

One of the issues a robot manufacturer faces is how to distribute programming commands/instructions/functions between the teach pendant, the terminal and microcomputer in the robot controller on the factory floor, and an off-line computer elsewhere. Much of the programming can be done off the floor, for example, input/output related to other equipment, and identifying (i.e., naming) the points the robot should reach in a sequence or path; the pendant is then used to establish the coordinates of these points. Some of the commands/instructions/functions in the categories mentioned earlier are programmed off-line, some at the floor terminal, and some at the teach pendant. A detailed analysis would surely show wide variations in locations between manufacturers.

Published descriptions in manuals reveal considerable variation also in the extent and the method of command abbreviation (one letter, two or three, more, none), in menu construction, and in error messages. An examination of the list revealed the following totals among the manufacturers: 42 (including information), 45, 124, 133, 201, 241 (including status), and 745. Though these totals are substantial, it is not clear in many instances the number of messages that are conveyed to factory-floor programmers nor how these messages are conveyed. It also isn't clear if help is provided for error recovery and if it is, how that help is provided.

DOCUMENTATION

User manuals for programming and robot operation also vary considerably among manufacturers in presentation and clarity. Some have gaps. Some must be difficult at least in part for inexperienced programmers to understand, even after they take training courses. The companies were helpful in providing information and manuals for preparing this report, and their cooperation has been much appreciated. But it was not easy to extract all the data appearing there. What would be useful for writing manuals, training courses, and equipment and software design would be straightforward and comprehensive descriptions of the programmer's performance--what the programmer does or should do with the controls, displays, and software associated with an industrial robot. In human factors engineering such procedural descriptions, which are standard undertakings in the development and use of many human-machine systems, are called task analyses. These appear to be largely unknown in industrial robotics, though one company's "tutorials" and another's "programming robots by example" are steps in that direction. So are another's task-oriented, performance-based video training packages. Another problem is how or whether to provide a sufficient overview about a process or procedures before going into details. A further difficulty has been to explain clearly in writing or diagrams the relationships between various coordinate frames of reference used in programming and how tool orientation and tool center point positioning are related. Another challenge has been the exposition of commands and instructions and an explanation as to where these are programmed. In reaction to such comments, one company's documentation manager made these observations:

Many of our manuals are written to give an indepth understanding of the robot only to those users that obtain a certain amount of actual hands-on experience. Our approach is intentional in that

most of our working manuals are written to the serious users that have purchased our robots and avail themselves of all our information facilities (reference manuals, programming guides, job performance aids, stand-up training, videotapes, self-paced modules, specific task application cells, etc.) Writing our manuals to the depth necessary to completely indoctrinate the potential user, the occasional researcher, and/or the casual reader that have not had hands-on experience and probably do not intend to have it has not been considered cost-effective.

HUMAN FACTORS INTERESTS

How much interest have robot manufacturers shown in human factors engineering? This has varied and has not resulted in a systematic introduction of that discipline. There have been a few ingenious innovations in pendant design and considerable attention given to grouping buttons by function. Color-coding and graphics have been exploited by some companies, but only to a limited extent. Two manufacturers have mentioned human engineering improvements (anthropometric or display/keyboard) in controller cabinets. One company stresses "man-machine communication" and another states it has used human engineering specifications (e.g., in color selection). Feedback from users has been mentioned by several as the basis of some redesign, though it is not clear if this has been sought in any systematic fashion. Software has in some cases been called "worker-friendly" or "user-friendly," with the use of English-like commands and menus, although no company seemed aware of research and applications in recent years to optimize the design of these for office systems (or to reduce hardware stressors in VDTs). Little heed seems to have been given to reducing software-occasioned errors and the methods of error recovery, or to help requests and responses. Software methods of imparting feedback seem to have aroused limited interest. Even though there has been considerable analysis (though not task analysis), the survey revealed only one test on pendant design with respect to speed or accuracy of use--an experimental comparison of a joystick and buttons for moving the robot in programming it. Apparently, there has been no such testing of any software or nonpendant hardware design features, yet "ease of use" and "user-friendly" need to be measured in some fashion other than the intuition of the designer. Data about performance speed in programming are important to productivity, and error data have major implications for robotic safety, since programming errors can lead to accidents incurring personnel injuries or equipment damage.

The dilemma regarding the allocation of programming between pendant and terminal apparently has not been construed as a matter for human factors investigation and for human factors support to system designers. In general, human factors engineering (or what it tries to do) has remained relatively unknown among robot manufacturers. Only one project has been reported describing the explicit application of human factors engineering to the design of equipment and software in human-robot interfaces in industry (Olex & Shulman, 1983; Shulman & Olex, 1985). Empirical data to guide such design must be sought either among factory users or in the laboratory. The latter seems more feasible.

PART TWO: GENERIC FUNCTIONS AND TASK ANALYSIS OF ROBOT PROGRAMMING

Section A: Outline of a Suggested Task Taxonomy

The outline that follows (pp. 17-28) is intended to list the kinds of tasks that may have to be undertaken on the factory floor or off-line to generate an applications program for an industrial robot. Several considerations must be noted.

The outline is limited in scope. It does not deal with operating systems, with aggregates of robots, or with extensive interfaces with other factory machinery.

Special emphasis is given to tasks involved in acquiring location coordinates with a teach pendant. Coding the steps in an applications program is not treated in detail. Not all of the tasks listed are to be found in the programming or operating manuals of all 11 manufacturers, although their publications were the primary source for the outline. Undoubtedly some tasks are missing, and many subtasks and task elements are not included.

As in many other performance analyses in human factors engineering, tasks are grouped into functions. There are six of these, which do not necessarily occur in practice in the order listed in the outline. For example, the function Sequencing Robot Motions/Actions may precede the function Specification of Each Robot Motion/Action. The function Conversion...Into Higher-Order Language (Coding): Writing and Applications Program may coincide with Sequencing Robot Motions/Actions. The functions Program Administration and Program Modification may occur in part during other functions. Similarly, the order in which tasks and subtasks are listed within functions does not necessarily reflect the order in which they are carried out in practice. Thus this outline constitutes a taxonomy more than a time-ordered task analysis.

SUGGESTED TASK TAXONOMY

- I. Preliminary Operations (at Robot Controller)
 - A. Powering up
 - B. Inserting disks
 - C. Loading software
 - D. Logging in: Identifying project and user

II. Specification of Each Robot Motion/Action

- A. Planning the destination of each motion
 - 1. Single point
 - 2. Index point for an array; other array points
 - 3. Index point for a path; other path nodes
 - 4. Point at a certain distance from a workplace
- B. Naming (declaring program position variables) each of these destination locations, indicating type of point
- C. Selecting the coordinate systems for identifying these locations in the program
 - 1. Rectangular (Cartesian) - X, Y, Z
 - 2. Orientation - yaw, pitch, roll
- D. Selecting the origin of the rectangular coordinates frame as a reference point
 - 1. Robot base (world frame)
 - 2. Wrist flange (tool frame)
 - 3. User-selected origin
 - 4. Origin by conversion
- E. Acquiring the coordinates of each destination location
 - 1. By entering them at a terminal from a point file, downloading them from another program, or deriving them from CAD data
 - 2. By "learning" them with a teach pendant
 - a) Preliminaries
 - 1) Calibrating robot axes (at controller panel or terminal), moving joints individually
 - 2) Establishing or checking software stops, moving joints individually
 - 3) Selecting initial reference point (origin)

- b) Enabling the teach pendant
 - 1) Connecting it to the controller
 - 2) Activating deadman switches
 - 3) Selecting teach/learn mode
- c) Selecting the type of robot movement by the teach pendant
 - 1) Along rectangular (Cartesian) axes (X,Y,Z) by arm joints
 - 2) Along angular axes (yaw, pitch, roll) by wrist joints
 - 3) About individual joint axes, each joint being controlled individually (often called "joint-coordinated mode," though this term refers actually to the mode for programmed motion)
 - 4) Free motion: No pendant control. The robot can be pushed or pulled about any joint axis
- d) Selecting the origin of the coordinates frame as a reference point for teach pendant movement
 - 1) Robot base (world frame)
 - 2) Wrist flange (tool frame)
 - 3) Other origin (user frame)
- e) Selecting the speed (velocity) of teach pendant movements (often called "jogging" movement)
 - 1) Between alternatives (e.g., slow, medium, fast)
 - 2) Faster or slower (e.g., "tick") movement
- f) Moving the robot to each planned destination location with teach pendant buttons or control stick (or by terminal commands)
 - 1) In position
 - 2) In orientation
- g) Precisely aligning the tool tip with the workpiece
 - 1) For fine positioning and orientation
 - 2) With close visual observation

- 3) Perhaps alternating between teach pendant and terminal
- h) Recording the position and orientation for each location
 - 1) Acquiring the name of the location at the pendant or at the terminal display (before sequence programming or after it)
 - 2) Activating the RECORD or WRITE button on the pendant
 - 3) Typing a command at the terminal
 - 4) Storing position and orientation for later use
- i) Modifying locations and coordinates
 - 1) Displacing an acquired position along a coordinate by a specified distance
 - 2) Finding distances between acquired positions or between positions and tool tip
 - 3) Moving tool tip directly to an acquired position
 - 4) Deleting a location name and its coordinates
- j) Replacing coordinates while initially teaching or during a demonstration or test run
- k) In approach submode, moving (with pendant) the robot toward a location until the display shows arrival

F. Specifying the programmed motion of the robot

- 1. Stating the trajectory type
 - a) Straight line
 - b) Joint-coordinated mode (curved; all joint movements end at the same time)
 - c) Other: Circular; continuous path; special type (e.g., weave)
 - d) Robot configuration: Right or left arm; elbow up or down

2. Stating the commands to move the robot
 - a) Move (to), Reach, Position
 - b) Approach; Withdraw
 - c) Orient
 - d) Transport
 - e) Expect (condition for next motion)
 - f) Move command is programmed by the record action at the pendant
3. Stating the robot's speed of motion (in execution)
 - a) Basic; default; selection; increment/decrement
 - b) Acceleration; deceleration
4. Specifying accuracy requirements (tolerances)
 - a) Coarse, fine, zerozone (corner-cutting)
 - b) Load limits
5. Specifying paths
 - a) Start and end points (nodes)
 - b) Modifications: Node deletions, node insertions; associated data
 - c) Replacing stopping with decelerations for a continuous path
6. Specifying arrays
 - a) Start and end points
 - b) Number of points and rows
 - c) Dimensions
 - d) Index command to increment
7. Specifying end effectors (tools)
 - a) Activate-deactivate (close-open)
 - b) Width of grip; amount of force
 - c) Tool dimensions

- d) Changing the end effector (next, previous)
- e) Modifying an end effector parameter during execution
- 8. Specifying inputs that will control robot motion/action
 - a) Delays: Values
 - b) Peripheral devices: Input signals (on, off)
 - c) Sensors: Values
- 9. Anticipatory triggering of execution
- 10. Specifying robot outputs to other devices
- G. Acquiring display assistance
 - 1. Information display (at teach pendant)
 - a) Current action data
 - 1) Command sequence number/current program line, step/point name
 - 2) Teach coordinate system in use for jogging
 - 3) Current mode for teach pendant
 - 4) Type of location: Single, index, or approach
 - 5) Joint values of location
 - 6) Coordinates of location
 - 7) Index pointer
 - 8) Teach velocity
 - 9) Tool at displayed point
 - 10) No location assigned to point
 - 11) Location out of reach
 - 12) Keyed input
 - b) Prior action data
 - 1) Number of points taught
 - 2) Sequences taught

- 3) Cycles completed
- 4) Last 32 binary signals (input/output [I/O])
- 5) Last 10/30 errors
- c) Future actions data
 - 1) Menu options
 - 2) Number of points remaining
 - 3) Cycles remaining
 - 4) Next step or position
 - 5) Subsequent positions
 - 6) Next 32 binary signals (I/O)
 - 7) Wait time
- d) Status
 - 1) Project number
 - 2) Program number
 - 3) Default settings (parameters)
 - 4) Type of unit (metric or English)
 - 5) Wrist model
 - 6) Close path assignments
 - 7) Software limit stops
 - 8) Electrical signals
 - 9) Limit switches
- e) Information feedback (at teach pendant)
 - 1) LEDs for button activations
 - 2) LCD for action undertaken
 - 3) LCD for responses to display requests (commands)
 - 4) LCD for programming errors

- f) Information acquisition
 - 1) Display requests: Data, lists, menus
 - 2) Menu level, previous menu
 - 3) Scrolling
 - 4) Cursor
 - 5) Windows (at CRT)
 - 6) Voice message
 - 7) Hard copy
- g) Instances of display assistance
 - 1) Menu: Action options displayed as prompts
 - 2) Specific information items
 - 3) Commands listed on a plastic card furnished to the user
 - 4) Commands and explanations in a manual (various)
 - 5) Tutorials provided in a manual for planning (1 tutorial), getting started (6), teaching and creating point files (8), creating program file (8), general (3)
 - 6) Tutorials provided in a program called Programming Robots by Example
 - 7) Performance-based training workbook
 - 8) Responses to HELP requests: (a) Prints a file of information that explains the topic you specify; (b) Produces a list of valid commands; (c) Displays and describes briefly each command, and then the first letter of the command, when typed, will get more information about it
- h) Display capabilities (teach pendant)
 - 1) Smallest LCD: 1 line, 12 characters
 - 2) Largest LCD: 8 lines, 40 characters each plus scrolling

III. Sequencing Robot Motions/Actions

- A. Generating a sequential flow of the items in IIB, IIC, IID, IIE, and IIF
- B. Specifying robot pauses or waits
 - 1. Time dependent (specifying time)
 - 2. Input dependent (specifying input)
- C. Specifying sequence changes by advancing to a later step
 - 1. Unconditional
 - 2. Conditional--dependent on some circumstance/signal
- D. Specifying sequence changes as interrupts because of input signals
- E. Specifying occasions for sequence terminations
- F. Specifying sequences described elsewhere
 - 1. Vision system
 - 2. Pallet organization
 - 3. Arc welding
 - 4. Peripheral devices
 - 5. Menu displays
 - 6. Error messages
 - 7. Help messages
- G. Specifying repetitions of commands, arguments, and location data
- H. Specifying repetitions of sequences
 - 1. Beginning and end
 - 2. Number
- I. Specifying decisions
 - 1. Comparisons between numbers (values, totals)
 - 2. Comparisons between discrete conditions
 - 3. Action alternatives in response to comparisons

- IV. Conversion of Robot Motions/Actions and Their Sequencing Into a Higher-Order Language (Coding): Writing and Applications Program
 - A. Naming the program
 - B. Writing successive steps as statements
 - C. Numbering (and labeling) the steps
 - D. Using the syntax of the particular language
 - E. Using the terminology of the particular language
 - 1. Commands and arguments/operands
 - 2. Declarations; identifiers
 - 3. Names of variables
 - a) Position variables
 - b) Nonposition variables
 - c) System variables
 - 4. Values of variables
 - 5. Subprograms (subroutines)
 - 6. Branching
 - 7. Jumps
 - 8. Loops
 - 9. Logical operators
 - 10. Relational operators
 - 11. Arithmetic operators
 - F. Indicating end of program
 - 1. For repetition of program
 - 2. For end of operation
 - G. Loading program disk into controller for interpretation/execution
- V. Program Modification (Testing, Debugging, Revision, Updating)
 - A. Revising location coordinates (teach pendant)
 - B. Revising program statements (editor or teach pendant)

- C. Executing program without robot motions
- D. Executing program with robot motions
- E. Simulating program input signals
- F. Testing with continuous execution
- G. Testing with single-cycle execution
- H. Testing with step-by-step execution
- I. Using the editor; various commands
- J. Receiving error notifications
 - 1. User errors
 - 2. System errors
- K. Coping with errors
 - 1. Error prevention methods
 - 2. Recovery procedures

VI. Program Administration (see also I. Preliminary Operations)

- A. Accessing programs
- B. Storing programs
- C. Duplicating programs
- D. Listing programs (directory)
- E. Deleting programs
- F. Modifying programs
- G. Creating files
- H. Accessing files
- I. Identifying, renaming files
- J. Listing files (directory)
- K. Transferring files
- L. Modifying files

M. Assigning project numbers

N. Assigning user numbers

O. Assigning passwords

Section B: Uses of a Performance-Oriented Analysis

The taxonomic analysis in Section A (pp. 17-28) is performance-oriented. That means it concentrates not so much on what the hardware and software are supposed to do, which is usually the focus of engineering and programming analyses, but on what the programmer should do so the hardware and software will perform suitably. Of course the hardware and software must be considered in the analysis because they are what the programmer must interact with, and in fact create or help modify.

The following subsections discuss the various uses to which the taxonomic analysis in Section A can be put, in industrial robotics.

There are nine ways that the taxonomic analysis in Section A can be used in industrial robotics. They are

1. **Design checklist** - A software or hardware designer can use the analysis as a kind of checklist to ascertain whether he or she has actually examined all performance aspects that might affect design. The taxonomy has been assembled from data related to a number of robot manufacturers. All the data items cannot be found in the documentation of any one manufacturer, at least those items that involve details rather than more general categories of performance. Though it is not assumed that all will be pertinent to each robot maker, a fairly comprehensive taxonomy may reveal, for designers, some gaps.

2. **Comparisons** - Together with the data base in the Appendix, the taxonomic analysis can support a comparative analysis of robotic systems to suggest elements of superiority from a human factors viewpoint. However, that has not been the purpose of this study, nor would it be justifiable; the source materials from any one company have not been complete enough and not all robot manufacturers are represented in this section or in the data base in the Appendix. Some comparative instances are presented for illustrational purposes (e.g., configurations, menu systems, error management, and display assistance) but the manufacturers from which these are derived are not identified.

3. **Programmer effectiveness** - Within the entire spectrum of software and hardware aspects of an industrial (or nonindustrial) robot aimed at making it effective are those aspects concerned with human-machine interfaces. These were the targets of this study. It has not tried to cover other aspects. In other words, its focus is on programmer effectiveness, not system effectiveness in general.

4. Design innovations - Though little has been done in industrial robotics under the labels of human factors engineering or ergonomics, this does not mean there have been no innovations that could be so categorized and that have been intended to make programming more effective. An effort has been made to list some of these innovations (see Table 3), both major and minor, if only to forestall reinventing them. Perhaps the most significant such innovation has been the substitution, in teaching destination locations, of moving the robot along rectangular (Cartesian) coordinates (with the teach pendant) rather than by activating individual joints (which has remained as an alternative). This significant development has been compared, for programming in general, to replacing assembly language by higher-order compiler or interpreter languages, or replacing machine language by assembly language--developments in the history of computer programming that could be called human factors innovations though they were not. The use of tool reference frames and user-determined reference frames in acquiring location coordinates has followed base-frame development and are considered innovations by some though apparently not all manufacturers.

Table 3

Particular Design Innovations That (May) Have Helped Programmer Performance

-
1. Teaching with pendant in rectangular coordinate frames instead of joints
 2. Use of tool reference frame and transformations to world frame
 3. Use of user-determined reference frames and transformations
 4. Single commands to move through an array of locations
 5. Adjusting robot position or sequence during an execution cycle
 6. Moving the robot directly to a taught point
 7. Calculating/displaying tool center point-taught point and point to point distances
 8. Incorporating weave motions for welding
 9. Identifying any untaught position found in a test run
 10. Making programming (sequencing) less complex
 11. Increasing programming coverage with greater sequencing complexity
 12. Adding a robotics supplement to a core higher-order language
 13. Distinguishing between program instruction categories
 14. Incorporating menus in programs
 15. Operator-created menus
 16. Stating commands in natural language
 17. Using single letters for terminal commands
 18. Combining robot motion and gripper opening in same command
 19. Combining teach movement and its speed in same button
 20. Replacing letters with symbols on hard pendant keys
 21. Progression through teach modes by repeated presses on same button
 22. Replacing hard pendant buttons with a control stick
 23. Replacing linear with logarithmic speed change in a control stick
 24. Color-coding of hard pendant buttons
 25. Putting windows in a VDT
-

5. **Neglected aspects** - Together with the data base in the Appendix, the task taxonomy has made it clear that many differences exist among the human-machine interfaces in robotic systems produced by different manufacturers, even though these differences have not all been spelled out. They raise two questions: (1) When there are alternative designs, which are better? Virtually no reliable (objective) data seem to be available on this question. It appears that robot manufacturers have neglected to try to ascertain, through systematic performance tests, the speed and accuracy of programmer performance with different design features, procedures, and languages. With one exception, the manufacturers have been content with proclaiming a feature as user-friendly, but without supporting evidence. (2) The second question concerns the effects that the variety of designs may have on programmers. When different interfaces for different robots exist in the same plant, might programmers become confused in shifting between them (should they be required to do so)? Such confusion could be reflected in higher error rates. This issue has rarely been raised in discussions about industrial robotics.

Another area of neglect has been outlined in II.G of the taxonomy, Acquiring Display Assistance. The items in II.G, pertinent to other components of the taxonomy in Section A, are drawn from most of the robot manufacturers surveyed. No single one includes them all, as becomes especially apparent from the fact that display capabilities may vary from one to eight lines and 3 to 40 characters per line. Which items are really needed by the programmer and which are not? But the more telling aspect can be seen in the modest extent of II.G's, Instances of Display Assistance, especially responses to HELP requests. Clearly this kind of display assistance, frequently provided in software for office or personal computer use, has not been well-recognized in industrial robotics.

Programmers make mistakes, as everyone knows. Though we don't know how many they make in programming robots, as noted earlier, many different kinds of errors can occur, as shown in Table 4. In addition to error prevention--a major objective of human factors engineering in robotics--programs can be developed that show a user (in this case a programmer) how to recover from an error, whether it is one the programmer made or one from other causes. If the recovery procedure is not displayed on a computer-associated screen, it can be set forth in a readily available manual. Although the survey in this study may not be complete in this respect, it indicates a general neglect of assisting programmers in recovering from errors.

One advantage of a task analysis is to reveal tasks or subtasks that have hardly been considered by designers because no control, display, or software is directly involved in the interface. In industrial robotics such a task is the visual observation in aligning a tool with a workpiece, with considerable precision. When carrying a teach pendant, the programmer must move very close to the alignment location so he or she can see whether the alignment is within allowed tolerances. As noted in Part One, this visual alignment performance is important because it is the very reason for having a teach pendant instead of using some nonportable control/display device. (It is also significant with regard to safety.) Yet it remains almost unmentioned in robot manuals and in the robotics literature. What proximities are

Table 4
Instances of Error Management

1. Totals of Error Types Listed in Manuals

- a. 201 errors
- b. 133 error messages
- c. 96 errors (among 241 error and status messages)
- d. 106 system fault and error messages (including 18 operational errors and error messages)
- e. 40 messages to indicate errors in command entry to the controller, as well as five categories of error codes: warning (nonfatal), with 477 items; pause (recoverable), with 27; aborts an executing program, with 128; cancel current motion, with 4; fatal (no recovery, must restart), with 97; and debugging, with 12
- f. 9 categories: warnings, programming (keyboard), hardware run time, machine recoverable, emergency, fatal, system software, and tape loading

2. Display of Errors

- a. An error shows on the display
- b. Display shows last 10 errors
- c. An error command in a menu allows viewing the last 10 detected errors
- d. A soft menu button shows the last 30 errors when repeatedly pressed
- e. Error codes are displayed, and a sheet provided with the equipment defines what they mean

3. Recovery from Errors

- a. A CLR ERR button clears the error
- b. Correct the condition that caused the error, and resume operations by pressing the RESET button (in a menu)

4. Prevention of Errors

- a. Both the RECORD and SHIFT keys must be pressed to record a location, and both the DELETE and SHIFT keys must be pressed to delete a location
-

required, under what illumination conditions, what equipment topographies, and what robotic operations? The only reference found, and it is very general, occurs in a Japanese research study of robotic accident prevention.

6. **New notions** - When a task analysis or taxonomy suggests a gap that has not been adequately considered, as the one just described, it becomes appropriate to propose a way of filling it in some alternative manner. For example, what might be done to support precise visual alignment that would not require using a teach pendant? As mentioned earlier, one notion is to use a CCTV system, mounting a camera on the robot's wrist with the television display at the computer terminal outside the robot's work envelope, so the programmer would not have to go back and forth from the terminal to the alignment location with a teach pendant. In addition to incurring hazards, such back-and-forth movement by a programmer has lengthened teach programming times (and costs) to the distress of robot-user manufacturers. Cameras have been mounted on robots for machine vision; perhaps the same ones could be used for human vision. Only a single, passing mention of such an approach has been found in the robotic literature, and it has never been carefully investigated. Its efficiency would depend, of course, on the required alignment proximities, as well as on other features such as illumination and contrast coloring or reflectance of the tool tip. These might also be optimized for precise alignment, which still depend on using a teach pendant.

7. **Desirable investigations** - In addition to the foregoing notions, a number of investigations, some major, some minor, seem desirable as a result of the task taxonomy/analysis--another use of this tool. These are outlined in Table 5. A large-scale candidate for investigation is the entire configuration for applications programming on the factory floor. Examples of configurations are summarized in Table 6. The issue in part is the previously noted division of effort between a teach pendant and a computer terminal; but related to this is the complexity of programming that is desirable--or feasible--on-line. The more complex, the greater the need for the VDT, because less of the total can be accommodated in the teach pendant, though there have been ingenious attempts to enlarge its capabilities (e.g., through menus and larger displays). Increasing a teach pendant's capabilities may increase its size and weight, thus making it more difficult or arduous to carry or manipulate. This question has clearly been a major design issue among robot manufacturers and manufacturing users. On the one hand, a robot manufacturer may produce a pendant with a larger display but fewer buttons thanks to a control stick and menus, and with less programming complexity than would be possible with a VDT (e.g., less reliance on relational and logical operators). This would accomplish all applications programming for some robotic operations. On the other hand, a different robot manufacturer may expand pendant capabilities but still rely on a VDT for much of the programming because of the increasing complexity of the workplace (e.g., peripheral devices and other robots). Though the issue has been joined, as pointed out in Part One, it apparently has not been resolved through systematic investigation unless such has been undertaken by manufacturing users; this study has not had access to those investigations. Factory-floor experience of any kind would be helpful.

An another important candidate for investigation is the design of menu systems. Examples are summarized in Table 7. Though the design of these has varied among robot manufacturers, complete details have not been found in the manuals available to this study. It is probable, however, that the relative advantages of various design features have received little analytic or experimental scrutiny, which is similar to the numerous investigations on menu design in human factors/ergonomics in recent years. The same may be said of other aspects of software design, such as command terminology.

Table 5

Design Aspects That May Need More Investigation

-
1. Speed and accuracy of programmer performance as a function of different design features, procedures, languages, and skill requirements
 2. Provision of adequate display assistance to programmers including information feedback and responses to HELP requests
 3. Methods of error management, especially programming errors and error recovery
 4. Tolerable complexity of applications programs especially with respect to relational, logical, and arithmetic operators and conditional sequencing
 5. Configurations of programming at a pendant and at a computer terminal with a nontextual or a textual programming language
 6. Coordination of the operations of a teach pendant and a computer terminal to minimize lost time
 7. Determining the visual proximities required in aligning the tool and the workpiece in various robotic operations
 8. Combining the downloading of CAD data and acquisition of location coordinates with a teach pendant--similarly for other off-line file data
 9. Methods for off-line programmers to access required data about robot operations to sequence these in applications programs
 10. Precedence in naming and sequencing robot location statements and in acquiring location coordinates

(continued on next page)

Table 5 (continued)

11. Programming paths and arrays of various types with appropriate indexing or changes in reference frames
 12. Programming or reprogramming during the execution of a robot motion cycle in a test or operational run
 13. Methods of revising programs (location changes, debugging) with various contingencies
 14. Design of menus with regard to paths through them, terminology, levels, and number of options per level, to minimize errors and time
 15. Terminology in commands, location names, and other data (including abbreviations) to minimize memory demands and confusion errors
 16. Relative advantages of reference frames with different origins (e.g., base and tool, absolute and relative points) and transformations between them
 17. Combining joint and rectangular movements in acquiring position or orientation coordinates with a teach pendant (buttons or stick)
 18. Methods of moving a robot to acquire location coordinates with a teach pendant: (a) buttons vs. stick; (b) simultaneous vs. successive buttons
 19. Speeds for moving a robot with buttons or a stick in acquiring location coordinates with a teach pendant
 20. Combinations of hard (dedicated) buttons and soft (menu) buttons in teach pendant programming
-

Table 6

Configurations*

-
1. The items in II.E's component on acquiring the coordinates of each destination location are managed with a teach pendant (TP), as well as several in II.F (specifying the programmed motion of the robot) such as the item on specifying end effectors. Otherwise, applications programming is managed at a VDT with a textual language.
 2. The items in II.E's component on acquiring the coordinates of each destination location are managed with a TP, as well as an item in II.F (specifying the programmed motion of the robot) on end effectors. Otherwise, applications programming is managed at a VDT with a textual language.
 3. The items in II.E's component on acquiring the coordinates of each destination location are managed with a TP. Some of the items in II.F (specifying the programmed motion of the robot), such as robot speed, arrays, end effectors, and peripheral devices, are managed either with a TP or at a VDT. Some of the items in I (Preliminary Operations) (e.g., file loading and program selection) are managed either with a TP or at a VDT. Other programming is managed at a VDT with a textual language.
 4. The items in II (Specification of Each Robot Motion/Action) are for the most part managed with a TP. Those in V (Program Modification) and VI (Program Administration) are managed either with a TP or at a VDT. Other programming is managed at a VDT with a textual language.
 5. The items in II (Specification of Each Robot Motion/Action), III (Sequencing Robot Motions/Actions), and V (Program Modification) were initially managed with a TP. More recently many or most of these (except those in II.E's component, acquiring the coordinates of each destination location) are managed alternatively at a VDT, together with those in IV (Coding), with a textual language.
 6. All applications programming is managed with a TP, with a nontextual language that includes relational, logical, and arithmetic operators.
 7. All applications programming is managed with a TP, with a nontextual language without relational, logical, and arithmetic operators.
-

*Refer to the suggested task taxonomy (pp. 17-28)

- Note.
- (a) These differentiations are actually more complex than they appear here; the aim is to present an approximate view.
 - (b) It should not be assumed that all the items in the task taxonomy are included to the same extent in each of these seven exemplars.
 - (c) Items in III (Sequencing Robot Motions/Actions) are managed with a nontextual language by TP key commands, including sequence changes, jumps, waits, and subroutines.
 - (d) Items in IV (Coding) are embedded in TP key commands for programming with a nontextual language.

Table 7

Menu Systems (Seven Robot Manufacturers)

- (1) No menus at TP or VDT
- (2) No menus at TP (first generation) or VDT
Menus planned at TP (second generation)
9 soft keys
- *(3) No menus at TP; some at VDT
No soft keys
Response by typing selection
Operator can create menus
- *(4) One menu designated as such at TP; no VDT (except optional portable one)
No soft keys
Access by: hard keys
Responses by: letter or numeral/symbol hard keys
Functions: additional commands in at least 14 categories
4 other TP hard keys display lists
Responses to lists by: letter or numeral/symbol hard keys
- (5) Menus at TP; no VDT
Bottom line on display: 5 possible menu options
5 soft keys under that line, one for each option position
Initial access by: 5 hard keys (top menu level); 2 other hard keys
Responses by: soft keys; numeral/symbol keys
Number of top level selections: 7
Number of levels: varying, mostly 3 or 4
Number of options within a level: varying, between 2 and 12 (5th soft key in a level displays more options in that level)
Functions: programming robot location data; programming nonlocation data; editing program; program execution; manual system control; vision system; arc welding
- (6) Menus at TP; none at VDT
Bottom line of display: 5 possible menu options
5 soft keys under that line, one for each option position
Cursor available
Initial access by: 5 hard keys (top menu level)
Responses by: soft keys; numeral/symbol keys
Number of top level selections: 5
Number of levels: varying, between 2 and 4
Number of options within a level: varying, between 2 and 5

(continued on next page)

Table 7 (continued)

- (7) Menus at both TP and VDT (data are for TP)
Bottom line on display: 5 possible menu options
5 soft keys under that line, one for each option position
Scrolling available
Cursor available
Initial access by: hard keys; menu display
Responses by: soft keys; numeral/symbol keys
Number of basic menus: 13
Number of levels: varying, between 1 and 5
Number of options within a level: varying, between 1 and 5
Functions: robot location data; controller operations; display

*Based on limited information

Note. Menus do not include dual-function hard keys with shift keys at TPs or VDTs.

8. **Skill requirements** - The allowable complexity of an applications program depends on the skill level of the programmer, a concern that robotic interface designers are aware of. Their problem, however, is (a) to relate skill requirements to particular aspects of program design, (b) to figure out how well a training program will establish the levels called for by such requirements, and (c) to establish, if possible, the availability of necessary programming personnel. This is a classic problem in human factors in system development. One of the uses of a task analysis or taxonomy is to break down the entire programming task into parts that have differing skill-level requirements. In turn, training programs can be tailored in terms of these; the task taxonomy may help ensure that no task component is neglected in training. If an adequate investigation of skill requirements and levels for programming on the factory floor has been accomplished, it has not come to our attention. Much, if not most, of the published attention given to robotics training seems to have been directed toward maintenance technicians and their training in special institutes, community colleges, or courses taught by robot manufacturers.

9. **Manuals and handbooks** - Some manuals for training have been published by robot manufacturers and have a performance orientation. That is, they describe what the programmer should do in creating and testing a program. Other manuals, whether for training or on-the-job support, are more likely to be object-oriented rather than person-oriented. They describe first the components of the software or hardware and then try to explain what the programmer should do with these. A case can be made that this direction should be reversed, as is done in a task analysis/taxonomy. One of the uses of a taxonomy is to help improve programming manuals in industrial robotics, not only by supplying a performance orientation but also by filling gaps and providing initial overviews. The fact that a few already have that orientation suggests that this proposal is not entirely novel.

CONCLUSIONS

The data base descriptions of pendants, terminals, controller panels, and software (see the Appendix) are evidence of the variety of ways these have been designed. The analysis summarizing these does not attempt to evaluate good and bad points. What is sorely needed is information from factory-floor users about problems they have encountered and errors that have resulted from various aspects of hardware and software design. The proposed task taxonomy could be useful in such a survey. Apparently, very little has been done by robot manufacturers to subject these interfaces to systematic human factors tests (under that label or not) or to document--at least for public consumption--feedback from users, though some has been received. Claims have been made that some units of hardware or software are user-friendly or easy to use, however, supporting data are missing. How and whether such data might be gathered within the robotics industry are questions for which no answers are presently apparent.

In view of this scarcity of empirical information in industry about the human factors effectiveness of various hardware and software design aspects, it will be possible to adduce technology transfer to interfaces in U.S. Army field robotic systems only through analysis based on general guidelines in human factors engineering. The Army's Human Engineering Laboratory itself will have to generate the needed empirical data, through a research program. That program, already planned as SRIP (Soldier-Robot Interface Program), is examining both mobile and stationary robotic units, including their control by teleoperation (telerobotics). Since such control resembles the teaching/programming operations for industrial robots, and the interfaces involved are similar to those reviewed in this report, its data base and task taxonomy should be of value to that purpose. However, there are also significant differences that must be investigated along with shared design problems.

Accordingly, the most reasonable approach for Essex Corporation to take in Phase II of this SBIR project will be to support the Human Engineering Laboratory in its SRIP research, by helping it establish both interior and exterior laboratory capabilities, suggesting appropriate research directions, and assisting in obtaining empirical data about human-robot interfaces.

REFERENCES

- Brantmark, D., Lindquist, A., & Norefors, U. G. (1982). Man-machine communication in ASEA's new robot controller. ASEA Journal, 55, 145-150.
- Department of Defense. (1981). Military Standard on Human engineering design criteria for military systems, equipment and facilities (MIL-STD-1472C). Washington, DC: Department of Defense.
- Galitz, W. O. (1984). Humanizing office automation. Wellesley, MA: QED Information Sciences.
- Gilbert, A., Pelton, G., Wang, R., & Motiwalla, S. (1984). AR-BASIC[®] An advanced and user-friendly programming system for robots. In Robots 8 Conference Proceedings: Future considerations (pp. 20-47 - 20-64). Dearborn, MI: Robotics International of the Society of Manufacturing Engineers.
- Grossman, D. D., & Short, W. M. (1985). AML-Much more than a robot language. In Robots 9 Conference Proceedings. Dearborn, MI: Robotics International of the Society of Manufacturing Engineers.
- Gruver, W. A., Soroka, B. I., Craig, J. J., & Turner, T. L. (1983). Evaluation of commercially available robot programming languages. In Proceedings of the 13th International Symposium on Industrial Robots and Robots 7 (pp. 12-59 - 12-68). Dearborn, MI: Robotics International of the Society of Manufacturing Engineers.
- Jacobs, M. P. (1984). Off-line robot programming: A current practical approach. In Robots 8 Conference Proceedings: Applications for today (pp. 4-1 - 4-11). Dearborn, MI: Robotics International of the Society of Manufacturing Engineers.
- Levosinski, P. E. (1984). Teach pendant control for robots. In Proceedings of the 1984 International Conference on Occupational Ergonomics. Rexdale, Ontario, Canada: Human Factors Association of Canada.
- Noro, K., & Okada, Y. (1983). Robotization and human factors. Ergonomics, 26, 985-1000.
- Olex, M. B., & Shulman, H. G. (1983). Human factors effort in robotic system design. In Proceedings of the 13th International Symposium on Industrial Robots and Robots 7. Dearborn, MI: Robotics International of the Society of Manufacturing Engineers.
- Parsons, H. M., & Kearsley, G. P. (1982). Robotics and human factors: Current status and future prospects. Human Factors, 24, 535-552.
- Salvendy, G. (1983). Review and reappraisal of human aspects in planning robotic systems. Behaviour and Information Technology, 2, 263-287.

Shulman, H. G., & Olex, M. B. (1985). Designing the user-friendly robot:
A case history. Human Factors, 27, 91-98.

Smola, P. (1986, August). Considerations in teach pendant design.
Robotics Today, pp. 17-18.

Stauffer, R. N. (1984, February). ASEA--its ambition is showing.
Robotics Today, p. 26.

APPENDIX

SURVEY OF ELEVEN ROBOT MANUFACTURERS' EQUIPMENT AND SOFTWARE INCLUDING A DESCRIPTION OF INTERFACE FEATURES

1. Adept Technology, Inc.	43
2. American Cimflex Corporation	48
3. ASEA Robotics, Inc.	56
4. Cincinnati Milacron	63
5. CIMCORP	75
6. GMF Robotics Corporation	82
7. International Business Machines Corporation	90
8. Schrader Bellows Division of Parker Hannifin Corporation	94
9. Toshiba Corporation	98
10. Unimation	103
11. Hitachi, Ltd.	111

SURVEY OF ELEVEN ROBOT MANUFACTURERS' EQUIPMENT AND
SOFTWARE INCLUDING A DESCRIPTION OF INTERFACE FEATURES

1. ADEPT TECHNOLOGY, INC.

TEACH PENDANT (Manual Control Pendant) (see Figure 1)

Displays

The display is a two-line liquid crystal display (LCD) with an adjustable viewing angle and LED indicators with some of the push buttons, as well as a row of five LEDs indicating the motion state of the robot in its Manual mode: WORLD, TOOL, JOINT, FREE, or FORCE (not used).

Controls

Forty-two elements (47 actions, not including those of menu soft buttons) include

a. Two (adjoining) buttons to stop the robot's movement, including a large one with PANIC (emergency stop) on it in red lettering.

b. Three (in a row) for mode control (all red): RUN/HOLD, STEP, and COMP/PWR (COMP/PWR turns arm power back on; COMP/PWR and RUN/HOLD have dual functions). These can also stop the robot's movement.

c. Three (adjoining, in blue) for moving the robot and selecting its speed. Two are graphical "speed bars." One moves a joint (or axis) in a clockwise (+) direction, the other moves a joint in a counterclockwise (-) direction. Where a speed bar is pressed selects the robot's speed. The third (toggle) control, SLOW, changes the speed range. The robot moves only when a bar is being pressed (e.g., by the left hand holding the pendant).

d. Three (in a row) including one REC/DONE (dual function) that is pressed to record the location of the robot's arm in the computer program for subsequent playback.

e. Three (in a column) function buttons not presently used.

f. Ten for numerals (standard number keypad). (These also control cursor movement for training parts of a vision system.)

g. Eight (in a column) to select among robot motions, all blue. They can select either (in the JOINT or FREE states) individual robot joints (by joint number) or among coordinate axes (in the WORLD or TOOL state), which the tool center point at the end of the arm and wrist should follow (with axis designators, X, Y, Z, RZ, RX, RY); thus, each button has two functions,

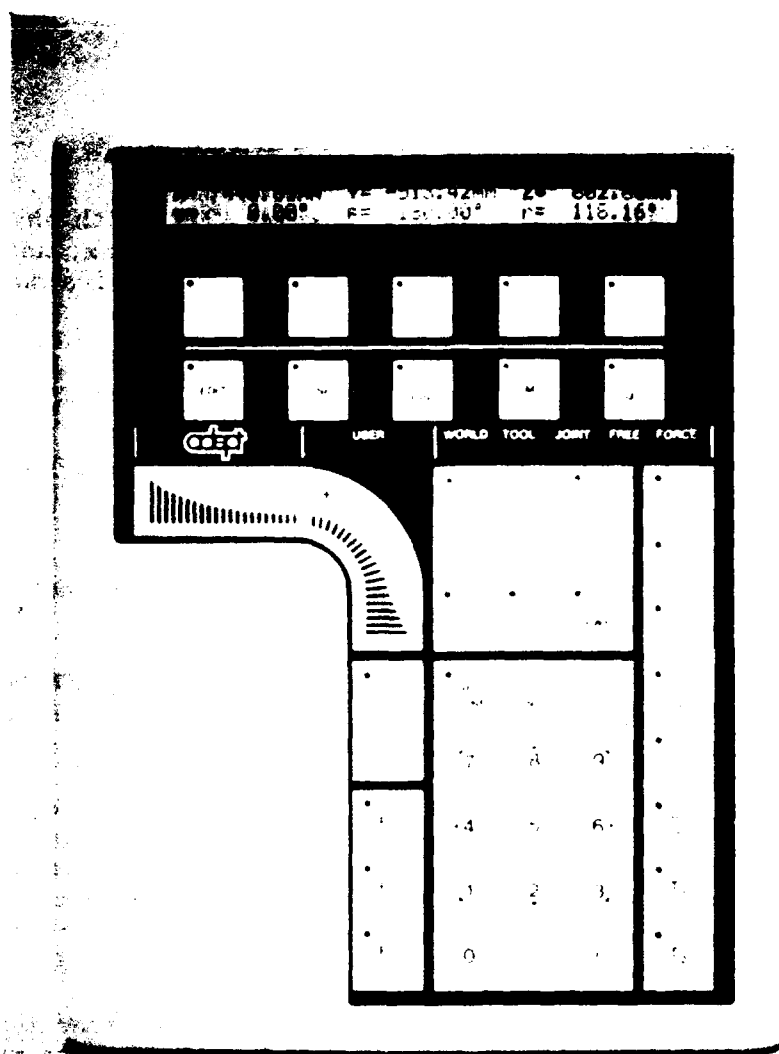


Figure 1. Adept Technology, Inc. teach pendant.
(Courtesy of Adept Technology, Inc.)

determined by the state. (Only four joints/axes are actually used for the Adept One 4-degrees-of-freedom robot.) Two buttons (one unused) control the robot's gripper.

h. Five "hard" buttons (in a row) control the following functions: EDIT, DISP, CLR ERR, CMD, and PROG SET. These constitute a top or primary menu level.

i. Above them is a row of five "soft" keys, unlabeled, just under the display, that designates what functions each can perform. The selection of one of the five hard buttons puts a second-level menu on the bottom of the display; each menu item indicates the function of the soft button beneath it.

There are five modes: Background, Uncalibrated, Arm Power Off, Manual, and Computer. The programmer proceeds from Background to Computer by pressing the COMP/PWR button, which also turns the arm power on, then the programmer can enter the Manual mode (to move the robot) by pressing MAN/HALT (next to the PANIC button). At first such motions are automatically in the World (coordinates) state; that is, the robot's joints operate together so it will move along Cartesian coordinates. Pressing the MAN/HALT button again further changes the state to Tool, in which joints operate together to move along tool coordinates. To enter the Joint state, the MAN/HALT button is pressed again, and now each joint can be operated by the pendant separately. Still another, successive activation of that button shifts to the Free state, in which each joint can be operated separately by direct manual guidance (as noted earlier, this is an option for some smaller assembly robots). Another press returns to the World state.

To move the robot within the World, Tool, or Joint states, the programmer must select the coordinate axis (by letter) or the joint (by number) among the robot motion buttons and then press one of the move/speed bars (thereby also selecting the speed). Primarily, teaching the robot's movements is done in the World (location) and Tool (orientation) states. When the tool center point (tip of the tool held by the robot's hand) reaches the position desired, with the desired orientation relative to the arm's X-, Y-, and Z-axes, the teacher/programmer records (stores) the position and orientation in the applications program by pressing the REC/DONE button. Then in a production run or other playback the robot will automatically go to them. With the Joint state, a position or orientation can be easily modified individually (e.g., for fine adjustment); also, a joint's electromechanical mechanisms can be tested by maintenance personnel. Programs are executed in the Computer mode, by pressing the RUN/HOLD button; pressing it again stops execution, and another press resumes execution. Pressing the STEP button stops execution after the current step in the program.

Control is also accomplished in the Computer mode by means of the menu system consisting of the display, five hard-function buttons, and five soft buttons mentioned earlier. For example, the hard DISP (display) button produces on the display, as labels for the soft buttons, Joint Values, World Location, Status, Binary I/O (input/output), and Last Error. A selection among these produces the indicated information on the display. For example, Last Error displays, in regressing order, up to 30 stored errors. The hard CMD (command) button enables a selection among commands for (a) loading a file from the mass storage device into active memory and starting execution,

(b) calibration, (c) storing from active memory into storage, and (d) executing the program automatically. The PROG SET button produces a menu for selecting a program, initializing one, priming it, and executing it. The menus accessed by the EDIT button permit entering new values at the numeral keypad and making changes with +/YES and -/NO buttons next to the REC/DONE button, for each location.

Design Aspects

Coding/Grouping

Control elements are coded by category rows, columns, or proximity, and by color (red, blue, and black/brown). Functional grouping characterizes joint select keys, program control keys, and number keys.

Labeling

Labels longer than four letters are abbreviated. Dual purpose keys show the two functions one above the other. Robot motion selection keys have numerals and letters, but no graphics. Robot movement/speed bars have graphics to indicate by analogy speed increase or decrease locations.

Feedback

For most control elements, an associated LED goes on or off. Hard and soft function keys show results on the LCD.

Errors

Previous (last 30) errors can be displayed through menus. Errors appear on the LCD, and an LED key labeled CLR ERR turns on.

Multiple functions

Twelve hard keys have dual functions, including six robot motion selection keys. Selection between functions for any key derives from program operation rather than a shift button or category selection button. Five soft keys have multiple functions, in a menu system in which the first tier consists of hard keys.

Miscellaneous

Labeled LEDs indicate which motion state is in effect; these are not introduced by separate, dedicated buttons but by the successive activation of the MAN/HALT button.

CRT/KEYBOARD TERMINAL (Computer Terminal)

With this unit the teacher (programmer) can, by typing them, duplicate commands from the pendant (except the coordinate axis and joint motions) and issue commands the pendant cannot. Requisite are the space bar, shift, cap lock, backspace, and delete keys. The letter S stops the display, Q restarts it. O stops output to the terminal and starts it again; Z aborts the program. There are 13 basic commands as well as 19 for programming; eight of the programming keys can be designated by single letters. In addition, the VAL II language has more than 200 key words (that can be abbreviated) available to expert programmers.

CONTROLLER PANEL (Controller Front Panel)

This unit has three switches: (1) A hardwired Arm Power Off push button turns off power and an associated light; pulling it out enables power but does not turn the power on by itself. (2) A three-position Key Lockout switch (a) passes control to a remote computer; (b) gives the teach pendant program control; and (c) gives the terminal program control--but (d) other functions remain common to the pendant and terminal. Program execution is allowed from only one source (a safety feature). Regardless of key position, all sources may always monitor status. (3) A System Power selection switch controls power to the computer inside the chassis; turning it off destroys nonsaved information; turning it on illuminates a yellow light behind it and, briefly, a red light for installed mass storage.

SOFTWARE

Adept Technology, Inc. is licensed to use Unimation's VAL II software. (See the section on Unimation.)

HUMAN FACTORS INTERESTS

Advertising claims that Adept robot systems are worker-friendly, simple to command, and easy to use. It further claims that "user-oriented software lets you use simple English commands and 'helps' you along step-by-step." Also, "a unique, portable manual control pendant makes job setup as easy as walking the Adept One through its routine, without any special operator training."

2. AMERICAN CIMFLEX CORPORATION

TEACH PENDANT (see Figure 2)

The teach pendant described here for the American Cimflex (formerly American Robot) Corporation is the second-generation version of the Merlin 3-6 axis, jointed-arm robots (see Figure 3). The first-generation version has encountered problems (a) in push-button modification (engraved labels in the current pendant), (b) in user-acceptance of a joystick, and (c) in jointly using the current pendant and the CRT/keyboard terminal with its fuller exploitation of the AR-BASIC[®] software language. Factory programmers have tended either to neglect the terminal or to spend time going back and forth between the pendant and the terminal. Users said the stick got in the way of reaching the buttons, tended to make them overshoot a position, and was likely to be damaged if the pendant was dropped. Aspects of the second-generation pendant include nine menu-oriented soft keys as function selectors; an eight-line, 40-characters per line display with windows and the display of menus; numeral keys; axis keys replacing the joystick; an alphanumeric keyboard; generic classes of commands; and greater compatibility with AR-BASIC[®]. One issue, according to Allen Gilbert at American Cimflex (personal communication 1986), is how much of a language and how many commands to embed in a pendant; and how to allocate limited space to control elements and a display that must be large enough to rely on the pendant as an effective programming device. This issue can be viewed as a general problem in industrial robot control.

Displays

A display at the top of the pendant has two halves. New commands inserted by the teacher/programmer appear in the lower half, as do prompts for arguments (abbreviations [three or more letters] and numerals) and options. The command moves to the upper half when a new one is inserted.

Controls

These consist of a joystick and a keyboard (with some switches).

a. The joystick on the lower part of the pendant moves the joints/axes of the robot. It has three degrees of freedom: left-to-right, front-to-back, and rotational around the stick's axis (twist). The declination of the stick controls the speed of the robot's movements relative to a switch speed setting between zero and ten. A motor power-on switch activates a red light. A selector switch assigns the joystick control of either the robot's arm joints and Cartesian world coordinates or the wrist joints and Cartesian tool coordinates. The selection within these categories --between moving the arm joints individually and moving the tool center point along rectangular coordinates intersecting in the robot base, and between moving the wrist joints individually and moving the tool center point along rectangular coordinates intersecting in the tool flange--is made with one

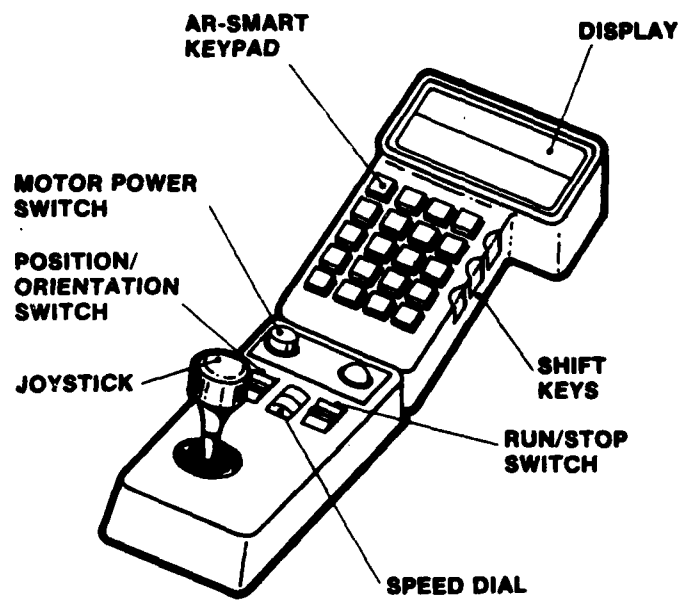


Figure 2. American Cimflex teach pendant.
(Courtesy of American Cimflex Corporation)

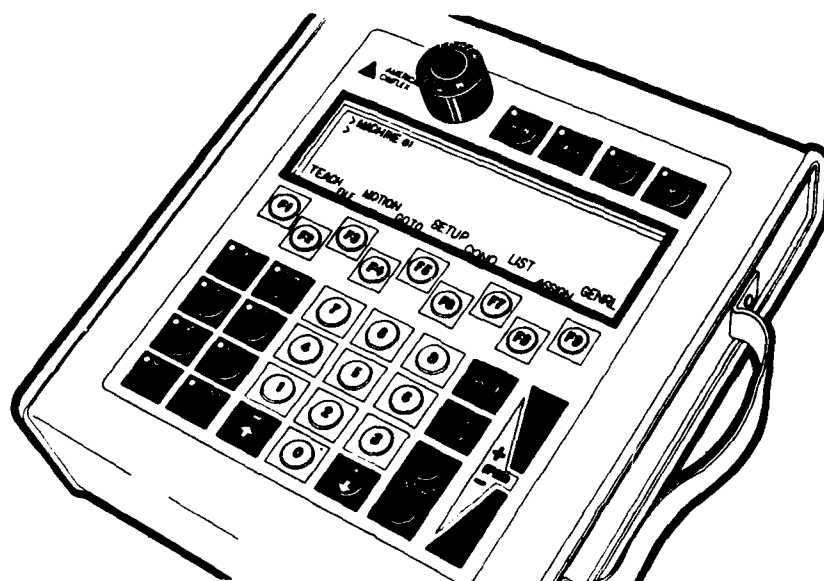


Figure 3. American Cimflex teach pendant: New version.
(Courtesy of American Cimflex Corporation)

of the push buttons on the keyboard. Thus the stick can be used for four options, each with three degrees of freedom: (1) moving the tool center point along the base-origin X, Y, and Z coordinates; (2) moving it along the tool-origin X, Y, and Z coordinates; (3) moving the three arm joints individually; and (4) moving the three wrist joints individually in pitch, yaw, and roll (around the tool center point).

b. The keyboard has 20 push buttons on it in a 5 x 4 array; one remains unused. Each button has multiple uses, designated by engraved labels for all AR-Smart commands--combinations of three (or fewer) letters (67 three-letter abbreviations), numerals, or symbols. Fourteen have five labels or uses, four have four labels or uses, and one has three labels or uses, for a total of 89 actions. The two top labels on the teach button are in black, the others are in red, blue, and green, in descending order. The items in black are a command and an argument (supplementing the command) or another command. They are activated without any manual selection procedure, and the computer program determines which is enabled. The red, blue, and green items--and actions--were selected in a prototype design by a sliding switch on the pendant's right side that could be moved from a normal neutral position (for black items) to a red, blue, or green position, to assign functions to the buttons. The current design has three shift keys, one for each color, that have to be held down while pressing a button with the same color on the pendant's face. Among the commands that can be delivered through this keyboard are eight for point definition (specifying the robot's position in space), four for tool and frame manipulation, seven for robot motion control, four for operational modes (immediate, learn, program), six for program logic, seven for arithmetic operations, six for registers, three for system status, six for file maintenance and system administration, twelve for program control logic, four for program execution, two for input/output, three for disk maintenance, and three for teach pendant control.

The American Cimflex Corporation's Merlin Robot System Operator's and User's Guide shows in detail, graphically, where each of these commands occurs in the 5 x 4 array and which color it has. It also has an alphabetical listing with the same information and advises users to photocopy this for ready access. The arrangements can be illustrated by some examples.

The three operational mode AR-Smart commands have the same button, top row, second column, with IMM (immediate in which the system powers up) in black, LRN (learn) in red, and PGM (program) in blue. The two robot speed commands (SPD and SMV) and the one to set software stops (SSS) also have the same button, second row, fourth column, black, red, and green. In the IMM mode, two of those for point definition (HPB and DPI) have the same button in the second row, second column--black and green. For moving the robot, MOV (moving the arm to a program-specified position) has a black label in the second row, first column. MVM (setting the movement mode, in both teaching and execution) has a black label in the fifth row, second column; its argument can be coordinate axis or individual joint motion in the LRN mode or straight line, joint-interpolated, or circular movement in the computer mode, represented by the digits 1, 0, or 2 (in black on the button's first row, second column; fourth row, third column; and first row, third column). STP (stopping continuous movement) has a red label in the second row, first column (same button as MOV). Two commands for defining tools (TDF and STT) have the same button, green and red, as do two for setting frames of reference (SFT and

FDF), black and blue. The command to define a relative point (HPT) (in coordinates relative to the current frame of reference) is found in red in the second row, second column, while that to convert to a relative point from an absolute point (REL) and that to convert to an absolute point from a relative point (ABS) are green and blue, respectively, on the same button, fourth row, second column.

To enter the program mode the programmer presses the PGM button (blue) in the first row, second column and to execute the program RUN presses the red button in the first row, third column. The programmer enters the immediate mode (IMM) at the first row, third column, black, and the learn mode (LRN), same button, red. Within the PGM mode, the commands for forward step, backward step, delete program step, and insert, all have the button in the fourth row, first column, with different colors. The same buttons are also used for if equal to (IFE - blue), if less than (IFL - red), if greater than (IFG - black), and if not equal to (IFN - green); for read input (IN - blue) and send output (OUT - red); and for SPG (save program), LPG (load program), and LPT (load point definition)--blue, black, and red, respectively. An adjoining button (black) in the next column is used for RNP (load and run program).

The black ENT (enter) key must be pressed after an argument but not after a command (when the other black command on the key, LST, for list, is enabled, listing the program currently in the system's memory on the diagnostic console). The ENT key also sequences through status items, gets the next screen, and resumes execution.

Design Aspects

Coding/Grouping

Control elements are coded or grouped primarily by color and share the same button, though there are some instances of coding or grouping by proximity.

Labeling

Labels consist of abbreviations (three letters or fewer), numerals, and typographic symbols. The 67 three-letter combinations have no duplications as such, though many share the same letter(s).

Feedback

Commands and their arguments appear on the display when entered.

Errors

These are communicated to the display (see the section on Software).

Multiple functions

This keyboard epitomizes multiple functions on hard keys, with 89 actions for 19 used buttons (keys), even though these do not include the robot movements performed by the joystick (and in most other pendants by buttons).

Miscellaneous

A joystick rather than buttons accomplishes the motions of the robot arm and wrist. Its three degrees of freedom are allocated to the arm or wrist by a switch next to it and are allocated to coordinate movements or individual joint movements by a mode selection button. (On the most recent design, the joystick has been replaced by movement buttons.)

CRT/KEYBOARD TERMINAL

A CRT display for the AR-Smart interpreter software and a keyboard are installed in the System Controller, which also houses the Controller Panel (see the next section). Before this recent addition, a video terminal and a printer were available as a diagnostic console.

New system software called AR-BASIC[®] has replaced AR-Smart but has incorporated some of its features, for example, the operation of the teach pendant. The CRT/keyboard terminal for AR-BASIC[®] has the following components:

1. Video screen has three windows: a top portion (the user can vary its size) for editing, with a blue background and black or yellow characters; a middle portion for commands, with a yellow background and dark characters; and a bottom portion for status information, with a black background and blue and yellow characters. A cursor (blinking black line) moves to the next character in the edit or command windows.

2. At the top of the keyboard there are 12 soft function keys for editing, designated F0 - F11. The functions of the last two have not been identified. Each of the F1 - F9 keys has two functions, one with and one without the activation of the shift key. The keys accomplish such editing tasks as moving the cursor in various ways, erasing characters or deleting lines, printing or reprinting lines, and placing the beginning or the end of a file in the edit window. Four arrow keys in addition can position the cursor anywhere in the edit window. Holding down a "command key" shifts from typing single letters to inserting 26 commands. Any "command expansion" is printed on the screen, along with the single letter on the key. The user can change the string of characters associated with each letter by using a SET-CMD-KEY command. Typing HELP and a command name prints a short file in the command window describing that command.

For other uses of the CRT/keyboard terminal in AR-BASIC[®], see the section on Software.

CONTROLLER PANEL (Front Panel and Operator's Panel) (see Figure 4)

1. The Front Panel has the following controls and associated displays:
 - a. An emergency off switch that cuts off all system power
 - b. A push button that turns system power on and a light on
 - c. A push button that turns on motor power
 - d. A key-operated switch (Reset-Run) that starts operations
 - e. A push button that overrides brakes on the robot
2. The Operator's Panel (optional) permits running a program without logging into the AR-Smart system.
 - a. A keyswitch selects the Run mode or the Program mode. A four-position thumbwheel selects the program number. The Run mode loads operating software from the system diskette. If the CAL (calibration) lamp lights, the robot is calibrated (it had battery-power backup); if the lamp flashes, the robot must be calibrated. A RUN button loads and executes the program. A STOP button stops it. A PAUSE button stops and restarts it. To log into AR-Smart, the programmer turns the keyswitch to the Program mode and presses the PENDANT button. A MANUAL switch enables the joystick. A lamp shows when the battery is low. An error lamp lights to indicate an error and is acknowledged by pressing an ERROR button.

SOFTWARE

As the name indicates, AR-BASIC[®] is based on and amplifies the BASIC language widely used as a high-level language in many contexts other than robotics. The AR-BASIC[®] system consists of a language interpreter, text editor, position definition library, motion control subsystem, and system manager; in addition, "MAGIX" is a real time UNIX[®] operating system. The AR-BASIC[®] system encompasses (a) a command mode - in which a robot is moved to a position, the position is (or has been) given a name, the interpreter stores the position's coordinates, an instruction file is created, and the program is executed; (b) an edit mode - for adding to and deleting from the file, only indirectly connected to the interpreter; and (c) a filing system with (1) a system memory (editor file and position definition library), (2) nonvolatile disk storage, (3) instruction files created by the text editor with file names, (4) position definition files created in part by the teach pendant joystick, and (5) general text and editor files. Applications programs can be created and edited by the text editor through the AR-BASIC[®] CRT/keyboard terminal described earlier. English-word commands are typed from the terminal. System commands include both general and file memory/management categories. The programming language includes both "core BASIC" commands and functions and "robotics extensions" commands and functions. Arguments include (a) parameters (e.g., on or off); (b) identifiers--names or labels the user creates, such as file names (information), variable names (scalars, points, tools, strings), statement numbers (at the start of each program line), or statement labels (identifying the lines); and (c) expressions (scalar, point, tool, string, and logical), built from constants, variables, operators, and functions.

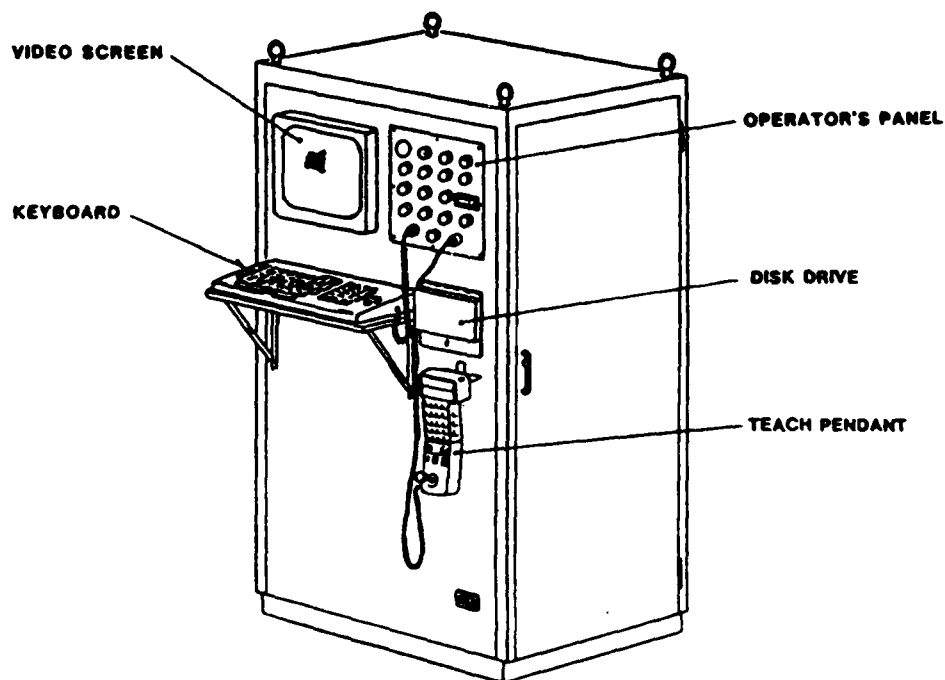
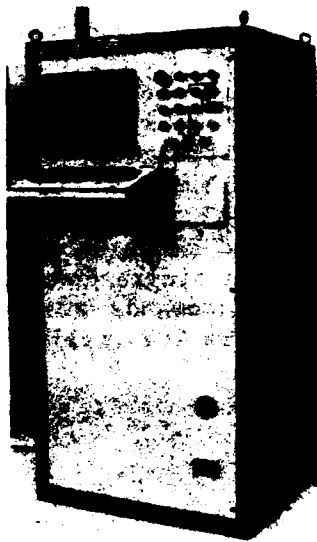


Figure 4. American Cimflex controller.
(Courtesy of American Cimflex Corporation)

A categorical listing of AR-BASIC® commands and functions sets forth the following categories: file, 13; editing, 5; manipulating scalar variables, 3; algebraic functions, 7; position definition commands, 7; position definition functions, 7; setting robot parameters, 6; motion commands, 3; program logic, 11; input/output, 10; execution and debugging, 6; and system commands, 9. The three robot motion commands are MOVE TO (point); ROBOT (ON/OFF); and WAIT-TILL-STOP. Various commands establish whether the joystick controls the robot in straight-line (coordinate-axis) paths or by individual joints, whether robot motion in execution follows straight, joint-interpolated, or circular motion, whether the robot's elbow should be up or down, and maximum robot speed. The LET command assigns values to scalar variables, frames, points, and tools. A new ARRAY MOVE command makes it possible for the robot to move through an array of positions with a single command. AR-BASIC® documentation contains 92 command descriptions; commands are spelled out, and some have abbreviations. In addition, there are 201 spelled-out error messages.

HUMAN FACTORS INTERESTS

An article by Gilbert, Pelton, Wang, and Motiwalla (1984, pp. 20-50, 20-51) concerning AR-BASIC® stated that "A limited pilot system was developed by March 1983. The pilot system was used to gain experience concerning human factors engineering. The pilot system was also used to evaluate robot programming requirements in the light of experience gained in the development of AR-Smart... AR-BASIC® is designed to present a highly 'user friendly' programmer interface by providing an extensive set of English-like commands (the commands in AR-BASIC®) in an information rich control environment." With respect to the CRT/keyboard terminal and function key-based text editing, this article said that "These interface elements have been designed according to tested human engineering specifications to achieve the maximum user benefits," (Gilbert, 1984, p. 20-51) and it lists as references a major human factors textbook and an article on color displays. According to Gilbert (personal communication, August 1985), the designers of AR-Smart and AR-BASIC® have greatly benefited from feedback from manufacturing users and factory-floor personnel. Even simplified programming has not seemed that easy for them, leading to the continuing evolution of the teach pendant. American Cimflex Corporation has employed a graduate student at Carnegie-Mellon University to provide support in human factors engineering.

3. ASEA ROBOTICS, INC.

TEACH PENDANT (Programming Unit) (see Figure 5)

Displays

The display can show 2 lines of 40 characters each. The top line contains messages. When one is longer than 40 characters, a SHIFT button puts

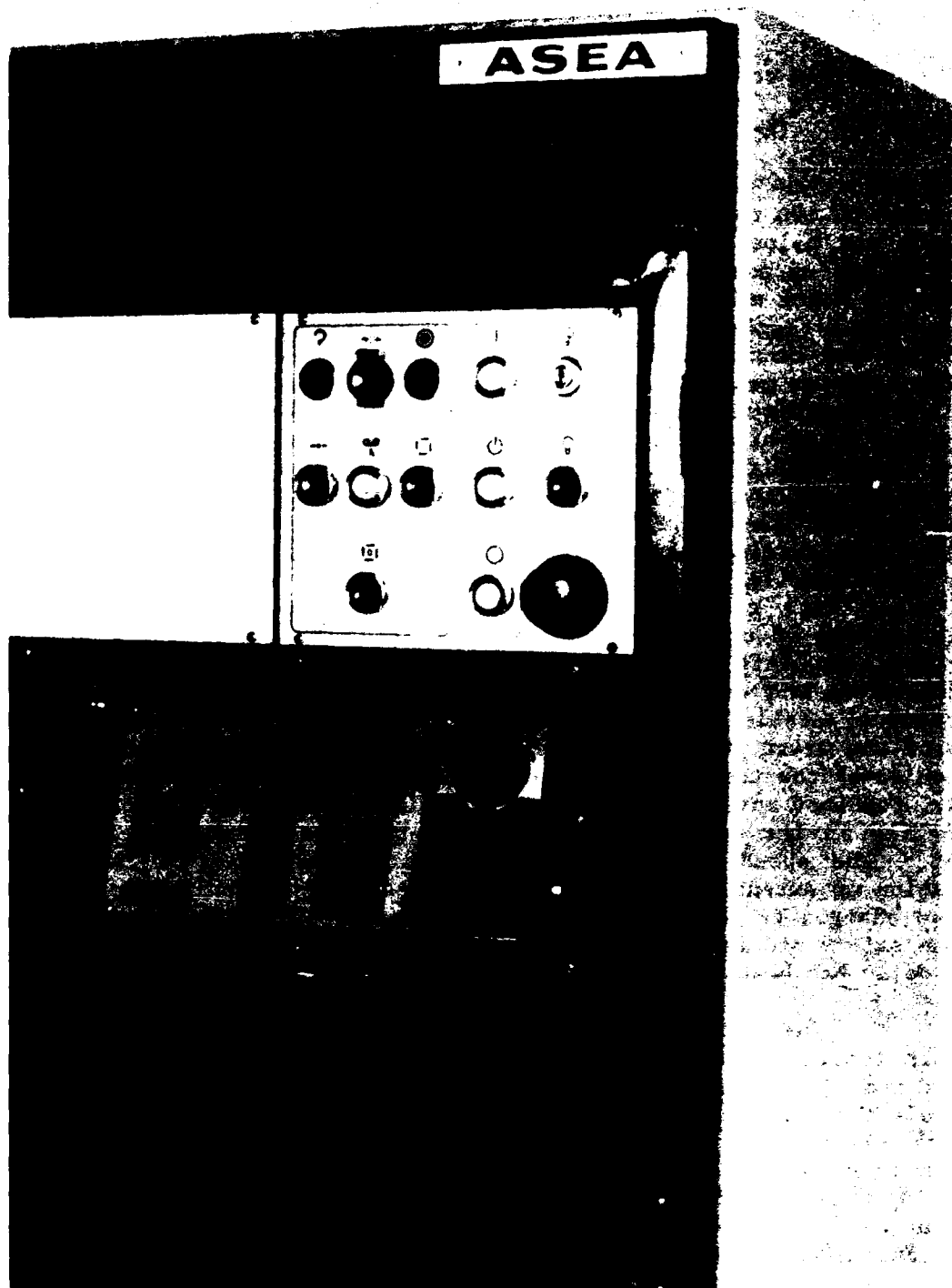


Figure 5. ASEA teach pendant and controller panel.
(Courtesy of ASEA Robotics, Inc.)

the overage on the line in place of prior characters. The lower line displays labels for each of the five soft buttons beneath it (in the menu system). Some of the hard push buttons have LEDs.

Controls

The controls consist of a joystick and a keyboard, plus an EMERGENCY STOP button projecting from the right side and a safety pad that functions as a deadman switch. The rectangular pendant's long axis is horizontal.

a. The joystick (controller) in the upper right corner is accompanied by two toggle switches, one to the left and above it, the other to the right and below it. They establish what the joystick does. The left toggle, in its up position, lets the stick move nonrobot devices; in its down position the stick moves the robot; labels consist of simple graphic outlines. When the right toggle is up, the stick orients the end effector on the wrist around the tool center point. When it is down, the stick moves the robot in a coordinate system mode. Labels are simply 2 and 1. In the coordinate system mode, both toggles are down. The system can be rectangular base-oriented, rectangular wrist-oriented, or cylindrical base-oriented, as selected by push buttons. In the first of these, stick movement forward and back (parallel to the short side of the pendant) moves the robot's tool center point (at the end of the tool on the end effector) parallel to the X-axis in the Cartesian (rectangular) coordinate system. Stick movement left and right moves the tool center point parallel to the Y-axis. Rotating the stick moves it parallel to the Z-axis, with clockwise rotation moving it down and counterclockwise rotation moving it up (like a screw). In the tool center point orientation mode, with the left toggle down, the right toggle up, forward and back movement moves the center of the wrist in a circular path around the tool center point parallel to the tool's Y-axis; side-to-side movement moves the center of the wrist in a circular path around an axis parallel to the Z-axis; rotation moves the tool center point circularly around an axis parallel to the X-axis. Combined movements are possible in both modes but are not recommended for jogging in the coordinate system mode because motions are difficult to predict. The speed of robot movements is proportional to the degree of declination or extent of rotation; initially it was directly proportional but has been changed to logarithmically proportional. The stick is enabled only when the safety pad is held down (by the side or heel of the hand); a 1-second safety delay has been added to forestall accidental stick activation.

b. The keyboard has 30 hard (dedicated) push buttons and 5 soft (multiple function) buttons. On the right a set of 12 buttons comprises a standard numeric keypad (with a period and a dash). Of the six in the middle, three select the coordinate system for robot motion, with diagrammatic labels. A fourth (labeled by three dashes) gives the robot its smallest possible movement with each press. Four others, with graphic labels representing a gripper open or closed, operate an end effector--a gripping tool or welding gun. The 12 buttons on the left consist of the following: (a) A stop button to stop automatic execution (but the joystick is still enabled and the pendant can still get program instructions); (b) Two speed correction buttons (+% and -%) to adjust the speed of the robot under program control in steps of 5 percent of basic speed, up or down; (c) A vision button (with the outline of a camera) for the machine vision system; (d) A process button (P) for arc

welding; (e) The shift key (mentioned earlier) that adds message overages to the display (and also clears error messages); and (f) Five function control buttons (with permanent diagrammatic or letter labels). One enables automatic program execution, either continuous or step-by-step; simulation of input signals; or displacement of previously programmed points. Another button (f) enables the entry of any program point except position. A third (with an arrow pointing down to a dot) enables the entry of position instructions and arguments. A fourth (f-g) places the system in the Edit mode. The fifth (with the outline of a hand) enables functions for manual operation and data entry. The menus that these can produce on the screen for the soft buttons are described in the section on Software.

Design Aspects

Coding/Grouping

Control elements are grouped into three spatially separated sets, according to categories of function: numeric keys, motion control keys, and other functions. Within the latter the six hard keys related to the display and menus are together at the top, with the soft keys above them under the display.

Labeling

Each key has a single item as a label: a numeral, a letter or letter combination, or a simple symbol or outline. The symbol or outline is intended to be an analog of what the button relates to or causes. ASEA robots are sold in many countries with different languages. Hence a design approach has been taken that is similar to that of international road signs.

Feedback

LEDs go on when some buttons are pressed. The display shows functions selected by the hard and soft function keys.

Errors

Error messages appear on the display.

Multiple functions

Other than the soft keys, only the joystick has multiple functions, allocated by the two toggle switches next to it and the three coordinate selection buttons.

Miscellaneous

A joystick rather than buttons accomplishes the motions of the robot arm and wrist. Its three degrees of freedom are allocated to the robot or

other devices by a switch next to it, and to coordinate or orient movements by a toggle switch on the other side. Push buttons select the particular coordinate system.

CRT/KEYBOARD TERMINAL

None is provided. All programming is accomplished through the teach pendant and control through the controller panel.

CONTROLLER PANEL (Control Panel)

This panel has the following controls and displays:

1. **MAIN POWER** switch - When it is turned to ON, a white light on the **SYSTEM OFF** panel goes on.

2. **SAFETY POWER** switch (an option) - This is actually on the side of the cabinet rather than on the panel. It can keep power out of the panel and can be interlocked with the **MAIN POWER** switch so both must be on to put power into the cabinet.

3. **SYSTEM OFF** push button - The label is a circle. This cuts power to the robot but leaves it in memory and the battery charger. It prevents execution, modification, or new instructions or programs.

4. **EMERGENCY STOP** push button - The label is a large encircled dot. This large, mushroom-type control stops all robot movement and turns on a red lamp as well as a lamp on the **STANDBY** button. The red light also goes on when there's an internal system fault. The emergency stop condition is removed by pressing the **STANDBY** or **OPERATION** buttons.

5. **STOP** push button - The label is a circle encircled by arrows. This interrupts (a) synchronization and (b) automatic execution of a program. It has no directly associated light.

6. **STANDBY** push button - The label is an upright bar across the top of a circle. This activates the robot's control system and makes memory available but does not provide operating voltage to the arm's motors; it lights a white lamp. It is ended by pressing the **OPERATION** button or **SYSTEM OFF** button or turning the **MAIN POWER** switch to OFF.

7. **OPERATION** push button - This is labeled by a vertical bar. When on, the entire system is available, except (a) if the robot and control system are out of synchronization and the synch lamp flashes, or (b) if no program is in the memory or one is invalid and the **FROM DISK** lamp flashes. It turns on a white lamp that goes off if (a) the **STANDBY** button or **SYSTEM OFF** button is pressed, (b) the **EMERGENCY STOP** button is pressed, or (c) the **MAIN POWER** switch is turned off.

8. **SYNCH** push button - This is labeled by a dash-dot-dash. It initiates the synchronization sequence. A yellow light is on during synchronization, but flashes when synchronization is absent. To resynchronize, the button must be pressed to make the light flash, then pressed again.

9. **FROM DISK** push button - This is labeled by two circles and a vertical downpointing arrow between them. It reads programs into memory from a floppy disk. A white light comes on for less than 1 second during reading and flashes if there is no program or the program is invalid during an attempted restart.

10. **START PROGRAM** push button - This is labeled by arrows encircling the digit 1. It starts continuous automatic execution of a program but cannot operate if the SYNCH or FROM DISK light is flashing. It turns on a green light that goes out if the STOP button is pressed or the teach pendant is operated.

11. A red **ERROR INDICATOR** lamp goes on when there's an operator error, system fault, or emergency stop. It has a ? as a label. The type of error shows in the teach pendant display. The lamp goes off when a correction is made or the SHIFT key is pressed on the teach pendant.

12. **LAMP TEST** push button - This, labeled by an outline of a light bulb, is pressed to test all lamps on the panel.

13. **PROGRAM LOCK** key switch - This has a key symbol as the label. It locks out the teach pendant completely or partially.

14. **BRAKE RELEASE** push button - This releases the brakes on the robot's arm, the system goes to **STANDBY**, motors lose power, and the arm may drift downward. Its label is a circle with horizontal arrows in both directions. It has a hinged cover.

Sequentially operated or functionally related controls and indicators are placed together; spatial separations prevent inadvertent confusion and activation.

Operations consist of (a) start-up (power on and synchronization); (b) a standby condition, in which (1) programs are read from a floppy disk to memory, (2) programs are written from memory to a floppy disk, (3) instructions are entered by the teach pendant, but without motor power, and (4) programs are edited; and (c) an operation condition, in which (1) programs are read from floppy disk to memory, (2) programs are written from memory to a floppy disk, (3) the joystick is operated to move the robot along coordinates, (4) instructions are entered into memory, (5) a program is test-run, (6) programs are edited, and (7) programs are executed (if there has been synchronization). Program execution may be continuous or step-by-step.

SOFTWARE

Instructions fall into the following categories:

1. Positioning instructions - After the joystick jogs the robot to the destination point, the POSITION button on the teach pendant registers the position of each axis in memory; then the required speed and accuracy are specified for execution.
2. Movement control instructions - These concern the speed, the tool center point, the coordinates, and the frame (coordinate system).
3. Peripheral control - of gripper and output.
4. Program control of the execution of instructions, programs, and program blocks (e.g., WAIT, JUMP, REGISTER, INTERRUPT, RETURN, GET, CALL-RETURN).

Operation of the hard function control buttons on the teach pendant may be illustrated as follows:

Pressing the button for automatic program execution brings the following functions for the soft buttons onto the display: PROG ST (continuous execution), INST ST (step-by-step execution), BWD (reverse step-by-step execution), SIM (simulates input condition), and SCAN, which presents additional functions: DISPL (display), VECTST, and ORIDE. Pressing the indicated INST ST button brings up a third menu level on the display for the soft buttons: GRIPPER, WAIT, OUTPUT, JUMP, SCAN, which continues the level with VELOC, CALL, RETURN, REG, SCAN, which further continues the level with TCP, INTER, COORD, GETB. Pressing COORD, for example, brings up a fourth level with ROBOT and RECT.

Pressing the hard function control key that enables the entry of position instructions brings to the screen for the soft buttons, VELOC, SAME, FINE, SEARCH, SCAN. Pressing FINE brings up, as the third menu level, LARGE, ZEROZONE, YES, NO. When a function is selected as the operator progresses through the menu, it is shown in the upper part of the display--in this case, FINE. Pressing the hard function control button for editing brings up V%, INSTND, STEP, MODPOS, SCAN, which continues with MODIFY, DELETE, INSERT, PROGRAM, SCAN. Selecting and pressing PROGRAM brings up PROGRAM NO.: CE, and ENTER. Then the operator presses numerals on the numeric keyboard to identify the program by its number and presses ENTER. Thereupon information about that program appears in the upper half of the display.

Documentation identifies 12 editing instructions/functions, 8 for positioning, 18 for control, 5 for automatic system control, 9 for manual system control, and 6 for adaptive control, as well as others for arc welding, links to superior computers, and a vision system. In addition, it describes 106 system faults and error messages in 30 categories and 18 operational errors and error messages.

HUMAN FACTORS INTERESTS

According to B. Weisbrodt (Stauffer, 1984), general manager of ASEA Industrial Robot Division, "In our new control, we stress man-machine communication by combining the use of a joystick with a self-instructing, interactive dialogue. The operator uses the joystick to run the robot manually, then handles the programming and other communication with the system via the alphanumeric display and function keys--in any of several different languages. We believe we have to make advanced technology easier to use every year." An experimental test by ASEA indicating the joystick was 25 percent faster than push buttons was reported in the ASEA Journal by Brantmark, Lindquist, and Norefors (1982).

4. CINCINNATI MILACRON

TEACH PENDANT (see Figure 6)

The present teach pendant is the second version over a period of 11 years, following two prototype units. According to R. E. Hohn (personal communication, August 1985), the first prototype unit incorporated three joysticks mounted in a box attached to a handgrip that was a helicopter control stick. It was rejected by users as too awkward, and the joystick approach was never tried again. The second prototype had push buttons instead of the joysticks. It was the forerunner of the first production version, which was used with the hydraulic robots introduced in 1975. That version (described later) had a minimal number of mechanical buttons. Associated with it was a built-in CRT/keyboard terminal at the controller outside the robot's work envelope. The pendant was designed to lead the robot to the points to be taught, and then, in replay, the required logic functions for these could be entered at the terminal. But programmers/teachers preferred to enter all the data for each point initially rather than during replay even though this procedure required them to return frequently to the terminal. Accordingly, when electric robots were introduced in 1983, a second-version pendant had many more buttons and a display to enable all the required operations to be performed at the pendant. The terminal was no longer needed. But to facilitate program debugging, there is an optional Portable Teach Station--a CRT/keyboard with a larger display and some dedicated command buttons but with fewer buttons than the pendant. It also omits those for moving the robot (programming its motions).

Displays

A relatively small 12-character display in the current (second version) (see Figure 6) pendant shows robot status messages. Long data messages are scrolled. Data to be displayed (prior to putting them in the program) are entered by an ENTER key. Various buttons can produce lists of available functions, velocities, tool dimensions, taught points, and data point attributes on the display. A MENU command key presents a list of additional commands, for selection.

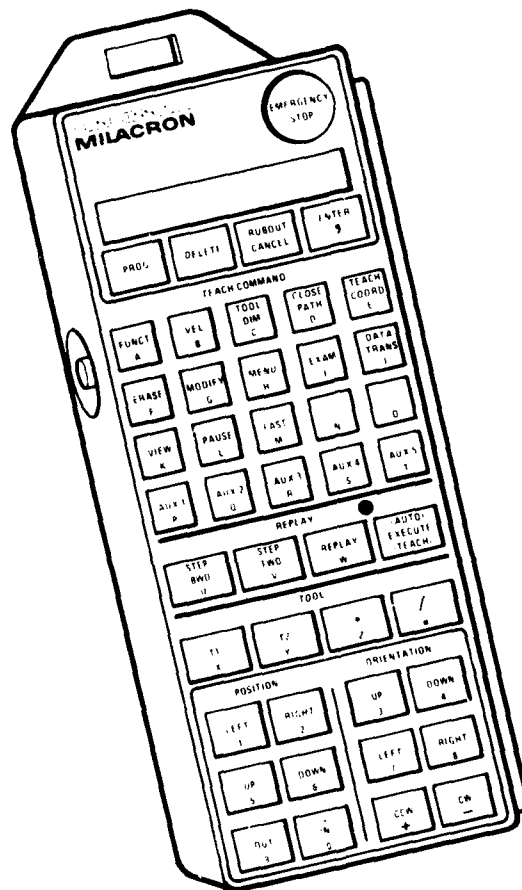


Figure 6. Cincinnati Milacron teach pendant.
(Courtesy of Cincinnati Milacron)

Controls

There are 44 membrane keys, as well as a large EMERGENCY STOP button at the top, a Deadman Handle (a bar) on the pendant's right side, and a Shift/Highspeed button on its left side. Releasing the Deadman Handle during arm movement by motion keys stops the robot and removes drive power through hardwired circuitry. To restart, the operator must depress the handle and then depress the ENTER key. While drive power is off, the robot is considered to be in a Power Pending state, which is displayed. The Shift/Highspeed button changes the functions of 39 of the 44 push buttons on the pendant's face from the function printed on the lower part of the button to that on the upper part. Thus, including the EMERGENCY STOP and Shift/Highspeed controls, the pendant has 46 control elements with 86 functions. The Shift/Highspeed control can increase the robot's speed (together with a motion key) in the manual and teach modes. The locations and functions of the 44 buttons on the face are as follows:

a. Four buttons under the display have six actions: (1) PROG, the program key, defines all data entered into the upper part of the display as the data of the point being taught; (2) DELETE deletes the point displaying where the robot is (and other points are decreased by one); (3) RUBOUT erases the last keyed entry (character). Holding this button down erases more until a comma is reached; (4) On the same key with RUBOUT, CANCEL clears errors or cancels data in the data entry area on the lower part of the display up to the last terminator (comma or ENTER); (5) ENTER enters data to be displayed, prior to using the PROG key; also, with the MODIFY command, it advances one data point in a sequence; and (6) On the same key with ENTER, a comma constitutes an intermediate terminator to separate teaching format entries.

b. Five buttons on the top row, in a Teach Command panel, operate as follows when the Shift/Highspeed button is pressed: (1) FUNCT (function) lists all available functions for a point, for selection; (2) VEL (velocity) lists all available velocities in a table, for selection or change; (3) TOOL DIM (tool dimension) lists assigned tool tables for selection or change; (4) CLOSE PATH closes the sequence just taught; and (5) TEACH COORD (teach coordinate) shows the teach modes available for selection (rectangular, cylindrical, and hand). These five buttons, bearing the letters A through E on their lower parts, also function as part of an alphabet keyboard.

c. Five buttons on the second row in the same Teach Command panel, with the keyboard letters F through J underneath, do the following in conjunction with the Shift/Highspeed button: (1) ERASE allows all taught points (a single one, sequence, or range) to be removed from memory; (2) MODIFY allows a whole job task, sequence, range, or single point to be modified; (3) MENU (as noted earlier) displays a list of additional commands available for selection; (4) EXAM displays a list of features that can be examined for selection and the status of the one selected; and (5) DATA TRANS transfers data between control and cartridge tapes or the host computer.

d. In the third row with the keyboard letters K through O are five more buttons on the Teach Command panel that operate as follows when shifted: (1) VIEW allows the viewing of data point attributes on the display; (2) PAUSE stops the scrolling of the display; this resumes when the button is released; and (3) FAST speeds up the scrolling.

e. In the fourth row with the keyboard letters P through T are five more buttons on the Teach Command panel. They are AUX (auxiliaries) numbered 1 through 5 for functions or command motion of optional equipment.

f. A Replay panel has four buttons (three with the letters U through W). With the shift, STEP BWD steps the robot backward to the previous taught point in the sequence; STEP FWD steps it forward to the next point; REPLAY moves the robot continually through taught points and executes functions (in the teach mode); and (AUTO) EXECUTE (TEACH) accesses the sequence in the teach mode to create or change it, or executes the function of a data point.

g. A Tool panel has four buttons (with the letters x through z and a period). T1 and T2 control tools; * controls multiplication and / controls division functions.

h. In the lower part of the pendant are 12 buttons for moving the robot and teaching it motions: six in a set for positioning and six in a set for orientation. In the unshifted state, the keys constitute a numeric keypad. In the shifted state, each button has one of two functions, depending on whether the teach or the manual mode has been selected at the controller panel. In the teach mode, the position keys labeled LEFT, RIGHT, UP, DOWN, OUT, and IN move the robot's tool center point along rectangular coordinates. In the manual mode they move the individual arm joints: base, shoulder, and elbow. In the teach mode, the orientation keys change the angles of the tool (and wrist) at the tool center point (which itself does not change). In the manual mode, the special three-roll wrist rotates up or down, left or right, and clockwise or counterclockwise.

The first-version pendant (see Figure 7) also used tool center point motion. For positioning, there were two buttons in a row for moving the base labeled L and R, two in a column for the shoulder labeled U and D, and two in a row for the elbow labeled OUT and IN. For orientation, there were two buttons in a row for yaw labeled L and R, two in a column for pitch labeled U and D, and two in a row for roll labeled CCW (counterclockwise) and CW (clockwise). That pendant had only 19 push buttons, including a Replay panel and a PROGRAM button, in addition to an EMERGENCY STOP and Speed control, and a selector switch. The selector switch had a keyboard (KB) position so a function on the CRT could be programmed with the PROGRAM button; a tool (T1) position so a tool was programmed with the current tool (T2) status setting with the PROGRAM button; and a C button for continuous program. The PROGRAM button could add or insert a data point in sequence.

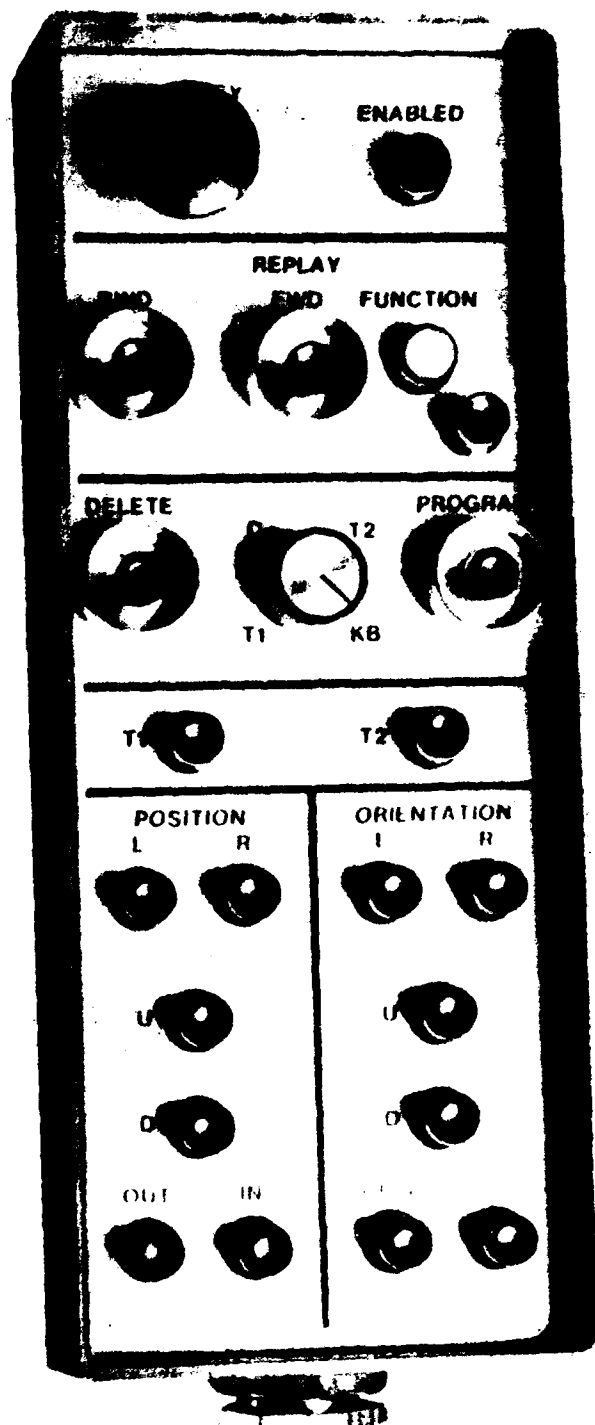


Figure 7. Cincinnati Milacron teach pendant.
(Courtesy of Cincinnati Milacron)

Design Aspects

Coding/Grouping

Control elements are grouped by categories in rows and blocks of rows, but categories are not spatially separated from each other.

Labeling

Identifying terms are spelled out except (in most cases) where the entire word would be more than six letters.

Feedback

Commands are shown on the display when they are entered. Buttons were given sensory deflection feedback after the initial design did not have it.

Errors

Error codes are displayed. A sheet provided with the system load tape defines what they mean (see the section on Software).

Multiple functions

Most of the keys have two functions (including letters and numerals), selected by a shift key, and each of the motion keys has three, selected by a shift key or a mode selector at the controller. To a limited extent, some hard command keys serve as second-level menu choices, with a listing of them on the display. A macro-option is available in which a customer overlay may be used. The macrofunction of the key is user-programmable.

Miscellaneous

The solution to the problem of using both a teach pendant and a CRT/keyboard terminal has been to enable all teaching capability into the teach pendant and have the CRT/keyboard terminal as an option. This optional unit has been packaged as a separate portable unit and may be moved from robot to robot.

CRT/KEYBOARD TERMINAL

As already pointed out, instead of a terminal outside the robot's work envelope, Cincinnati Milacron robots presently have an optional portable CRT/keyboard (portable teach station) that can be brought into the envelope along with the teach pendant. Otherwise, it rests outside on top of the

control cabinet. Its 15 dedicated command buttons have the same functions as the ones on the pendant but other commands/functions are typed; it also lacks the controls for moving the robot's tool center point or joints in position or orientation, as well as View, Pause, Fast, and Replay buttons. In addition to the push buttons, the keyboard has a numeric and punctuation keypad and an alphabetic letter keyboard (with keys for Cancel, Rubout, Comma, and Enter).

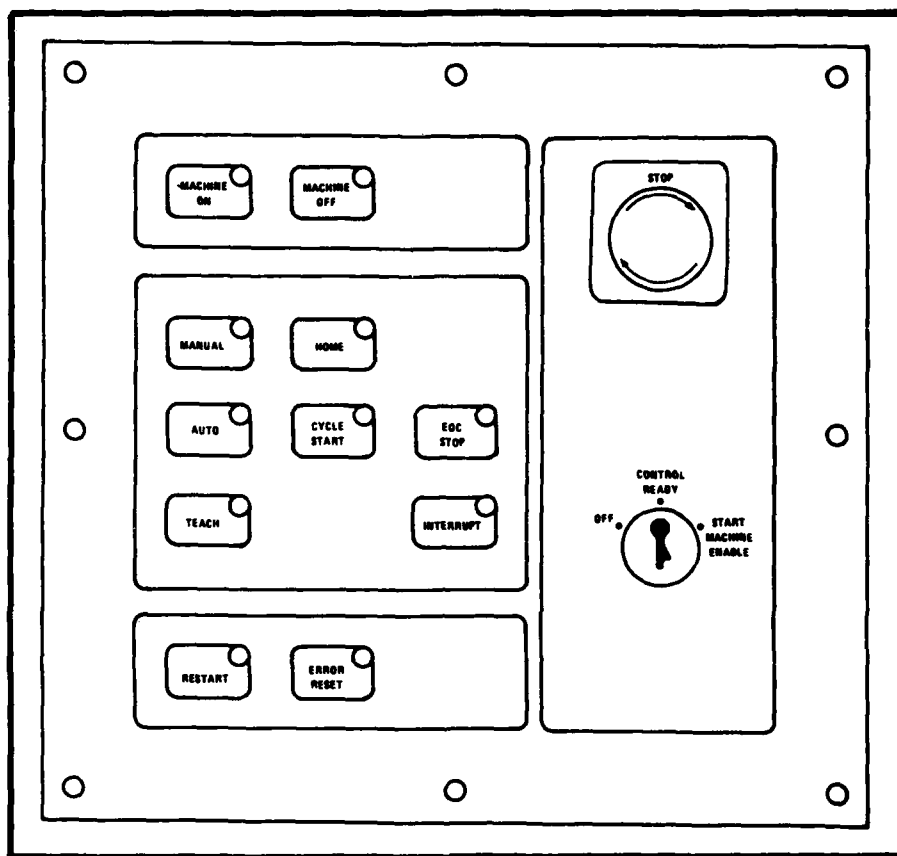
The screen has two sections. The upper half, the point data area, shows actual data entered and/or programmed. A point number and AT mean the robot is at those coordinates; FROM means the robot is away from that point. The lower half, the data entry area, displays data as it is being entered. "Command in progress" is shown when the teach pendant is entering data.

The earlier CRT screen had six display areas for (1) velocity factors; (2) physical location of an AT or FROM, and modify point, range, or sequence; (3) error codes and data entry; (4) functional data on a specific point--function, velocity, tool dimension; (5) physical location of the tool center point in X, Y, and Z in millimeters or inches (either may be selected at system definition time) and wrist orientations in degrees; and (6) the coordinate system of the teach mode--cylindrical, rectangular, or hand--and tool status.

CONTROLLER PANEL (Operator Controls) (see Figure 8)

The current version of this panel has the following elements (see Figure 9 for an earlier version):

1. A large EMERGENCY STOP button, when pressed, immediately removes drive power through a hardwired circuit. Axes are held by brakes.
2. A control rotary key switch (removable key) can be turned to OFF or CONTROL READY with a momentary turn to START MACHINE ENABLE.
3. Two push buttons operate in the Manual mode. MACHINE ON sends power to the axis drivers while START is held, and MACHINE OFF terminates it.
4. A MANUAL push button puts the robot into the Manual mode, in which the teach pendant can move the individual joints. (There are three modes: Manual, Teach, and Automatic.)
5. A HOME push button moves the arm to the defined home position, while pressed in the Manual mode. In the Auto or Teach mode it moves the robot from the cycle start position only via taught points, even if the button is released.
6. An AUTO push button puts the robot in the Auto mode. The robot can run automatically through the task.
7. A CYCLE START push button, while the robot is in the Auto mode, starts an automatic cycle.



RD-279-1

Figure 8. Cincinnati Milacron controller panel.
(Courtesy of Cincinnati Milacron)

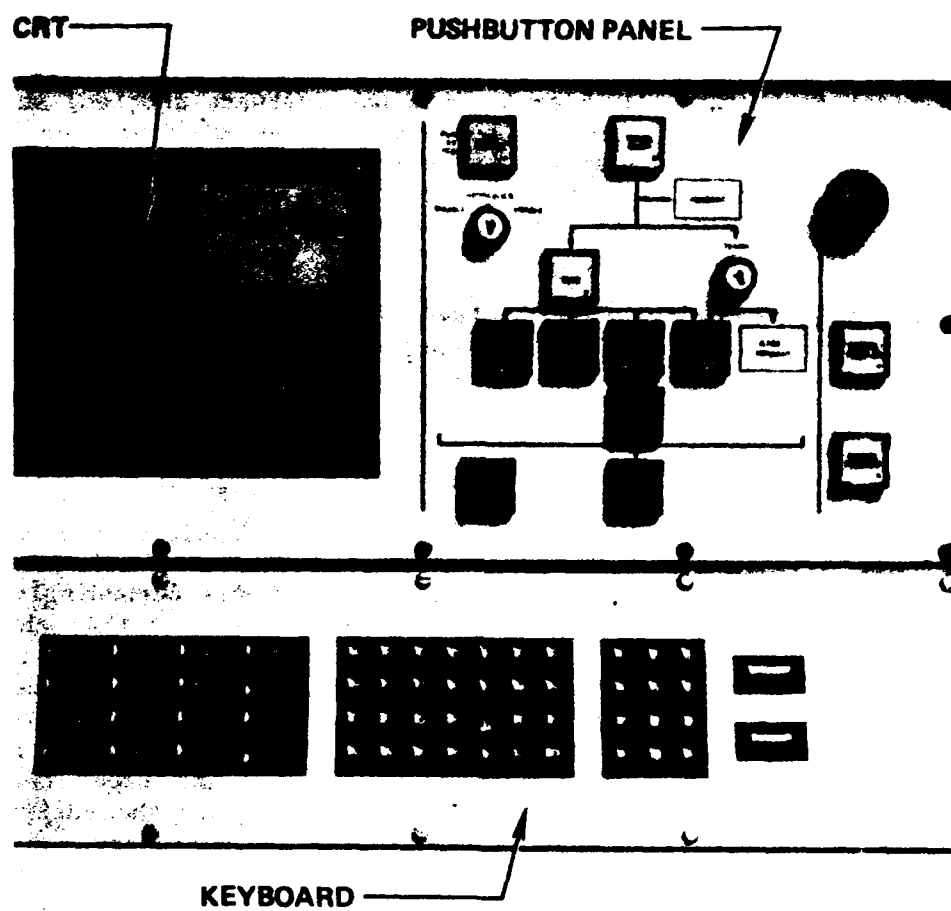


Figure 9. Cincinnati Milacron controller panel (earlier version).
(Courtesy of Cincinnati Milacron)

8. An EOC (end of cycle) STOP push button stops the automatic cycle when the robot reaches the cycle start point.

9. A TEACH push button puts the robot in the Teach mode (to teach tasks). A password-protect feature may be used.

10. An INTERRUPT push button stops the automatic cycle in process. All motions and functions stop. To restart, the operator presses the CYCLE START push button but any function in progress is aborted.

11. A RESTART button returns the arm to the last calculated position after the reset of an M, R, or E error (see the Software section) occurs with the robot on. To align the arm to restart, the operator presses the ERROR RESET button and restarts the drives if it is required (depending on the type of error) with the RESTART button, holding it down until the arm stops moving and the LED stays on; then the operator puts the robot in cycle by pressing the CYCLE START button.

12. An ERROR RESET button clears conditions producing E, R, or M errors and puts control back into the Manual mode.

SOFTWARE

The Cincinnati Milacron programming system "provides the user with features that resemble a language," according to Gruver, Soroka, Craig, and Turner (1983), "but it is not a textual language" and accordingly was not described or evaluated by those authors. However, the fact that the current pendant makes use of alphanumeric inputs suggests that this characterization may not be true of the current programming system. It has four categories of commands, plus a set of decision conditions and nine types of error codes: warnings (W), programming (P), hardware (H), run time (R), machine recoverable (M), emergency (E), fatal (F), system software (X), and tape loading (L). Commands can be entered through the keyboard by keying in the entire word or only a number of characters to distinguish it from another entry.

Teaching Commands

a. Teach coordinates: Coordinates are cylindrical, rectangular (the tool center point moves in, out, L [left], R [right], up, down), or hand (the tool center point moves relative to the wrist's face plate).

b. Tool dimension: This is keyed in inches (or millimeters).

c. Velocity: Speed can be set to any of 15 velocities or to none. These velocities may be factored up or down by using a velocity factor.

d. View: The pendant shows the attributes of the data point being displayed or modified (nine items).

e. Close path: This controls the close path sequence.

f. Erase: (various conditions)

g. Data transfer: (various conditions)

h. Modify: In the normal mode, this displays all necessary data to modify a single point, range of points, complete sequence, or all points in a program. In the screen mode, it displays points in a tabular format. Four keys move a cursor on the screen; Enter displays the next page of point information, and a comma displays the previous page.

Function Commands

A function command selects the function to be taught to a data point. A function is the action to take place at a point; every point requires a function. In the Auto mode, the function will be executed when the robot reaches the programmed points. In the Teach mode, the Execute (auto) key performs the function on a point. Points without operations to perform are taught the no-operation (NOP) function. Other functions include

Delay:	The robot stops at the taught point for a specified time; the operator keys in the seconds
Output:	Signals to be keyed are off (-), on (+), inverted I, toggled T, and output number
Wait:	The operator keys in the signal number for the wait; a default timeout value and an abort sequence may also be programmed
Perform:	To separate the task into parts or segments, the operator keys in sequence numbers; this number may be a variable
Disable:	To disable an interrupt or event, which has previously been enabled
Flag:	This is a user-accessible software register
Index:	This changes the entry point to a sequence
Variable:	This establishes the values of a system variable
Velocity:	To make a temporary change of an assigned value
Search:	Executes the function of the next sequential point after an interrupt
Position:	Stores into a specified data point the tool center point coordinates of a point when executed
Time:	Timers
DAC:	Digital to analog conversion

Weave:	To teach weaving motion (e.g., in welding)
Message:	Displays a message
FWP:	Programmable Workpiece Positioner
SWP:	Servoed Workpiece Positioner
Align:	To allow adjusting, with the pendant, a position during a cycle
STRK:	To follow the weld using a seam tracker
Vision:	To sense parts, identify them, determine their locations and orientation
Remote:	To send information to and receive information from the host computer
Restore:	To send specified outputs back to the last known state
Frame Align:	Sensor interface to provide information to shift the program in all six dimensions

There are also 16 Examine commands, to display features available in the controller, and 17 Menu commands. In the Teach mode, additional available commands are displayed. In the Manual mode, Set-Up, Normal, Home, and Error commands are accepted. In the Automatic mode, DAC, VEL, STRK, and Weave parameters can be changed. Menu commands include Copy: Allows taught paths to be repeated in other parts of the program; GOTO: With the pendant's STEP FWD (forward) key, the operator can move the tool center point directly to selected taught points from the current position; Mirror: Mirrors an entire task, sequence, range of points, or point, on the opposite side of the XZ plane; Distance: Calculates and displays the distance between the tool center point and a selected taught point; Pendant: Selects between the long and short mode of the teach pendant; for the long mode, for each command input there's a complete list of selections that scroll by on the pendant display, while for the short mode there is no list and the operator inputs the function; Home: Allows changing the home position; Weave Aid: Used to determine the azimuth and angles of weaves; Voice: Turns the message voice feature on or off; Trigger: Allows the taught function to be executed before the point is reached; Hardcopy: Produces a written record of the taught program; SysDef (system definition): Allows changing system definition parameters in the teach mode; Error: Allows viewing the last 10 detected errors; Operator Intervention: Used to modify some active parameters in the auto cycle; Pathlength (override): Enables a keyboard-entered distance; and Calibrate: Involves translating the external system to the robot's coordinate system.

Decision conditions include single conditions (on or off), multiple conditions (state of a combination of signals [AND, OR]), numeric comparison conditions (true or false results of EQ, NE, GT, GE, LT, or LE), and continue requirements, for example, the robot may continue through a point (always, never, option).

HUMAN FACTORS INTERESTS

R. E. Hohn (personal communication, August 1985) has pointed out that the need for feedback from users, the effects of design changes in the workforce, and the differences between how hardware and software are planned and how people actually use them, are all evidenced in the changes that have occurred in the Cincinnati Milacron teach pendants and CRT/keyboard terminals. For anthropometric considerations, the company has employed an engineer familiar with human factors engineering, in its Machine Tool Group, for two decades.

5. CIMCORP (formerly GCA Corporation)

TEACH PENDANT

The teach pendant is for the CIMROC[®]2 (Computer-Integrated Manufacturing Robot Control)-controlled DKF-200 6-axis robot only (see Figure 10).

Displays

These consist of a character display and LEDs. The display, which can hold 20 characters, has three horizontal sections: the upper one for the coordinate system selected--base (BS), joint (JT), and tool (TL); the middle section for point names and types, with \$ indicating the base coordinate system and @ indicating the joint coordinate system; and in the lower section the mode of operation--INDEX, SINGL (teaching a Cartesian point), and APPR. The display can also show six manual mode error messages.

Controls

The pendant has 28 buttons for 43 commands and functions. The upper four rows are predefined control keys and the lower three are drive keys.

a. Top row: (1) A BASE button places the robot and controller in the Cartesian coordinate system; BS shows on the screen. (2) A JOINT button places them in the joint coordinate system in which any joint can be rotated individually; JT shows on the screen. (3) A TOOL button places them in the tool coordinate system, with the middle of the tool flange as the XYZ intersection point; TL shows on the display. (4) A WRITE button defines and optimally declares points; it assigns values to point coordinates, that is, it assigns the current arm position coordinates to the current point coordinates. The first three buttons operate as selection devices for the robot motion (drive) buttons in the lower three rows on the pendant face. The fourth is affected by the INDEX POINT/SINGLE POINT key. In indexed point operation, a new point has the same type and name but its index is incremented by one.

b. Second row: Two buttons activate and deactivate tool keys, for two tools each. The left key activates tool 1 (ACT.T1) or tool 2 (ACT.T2),

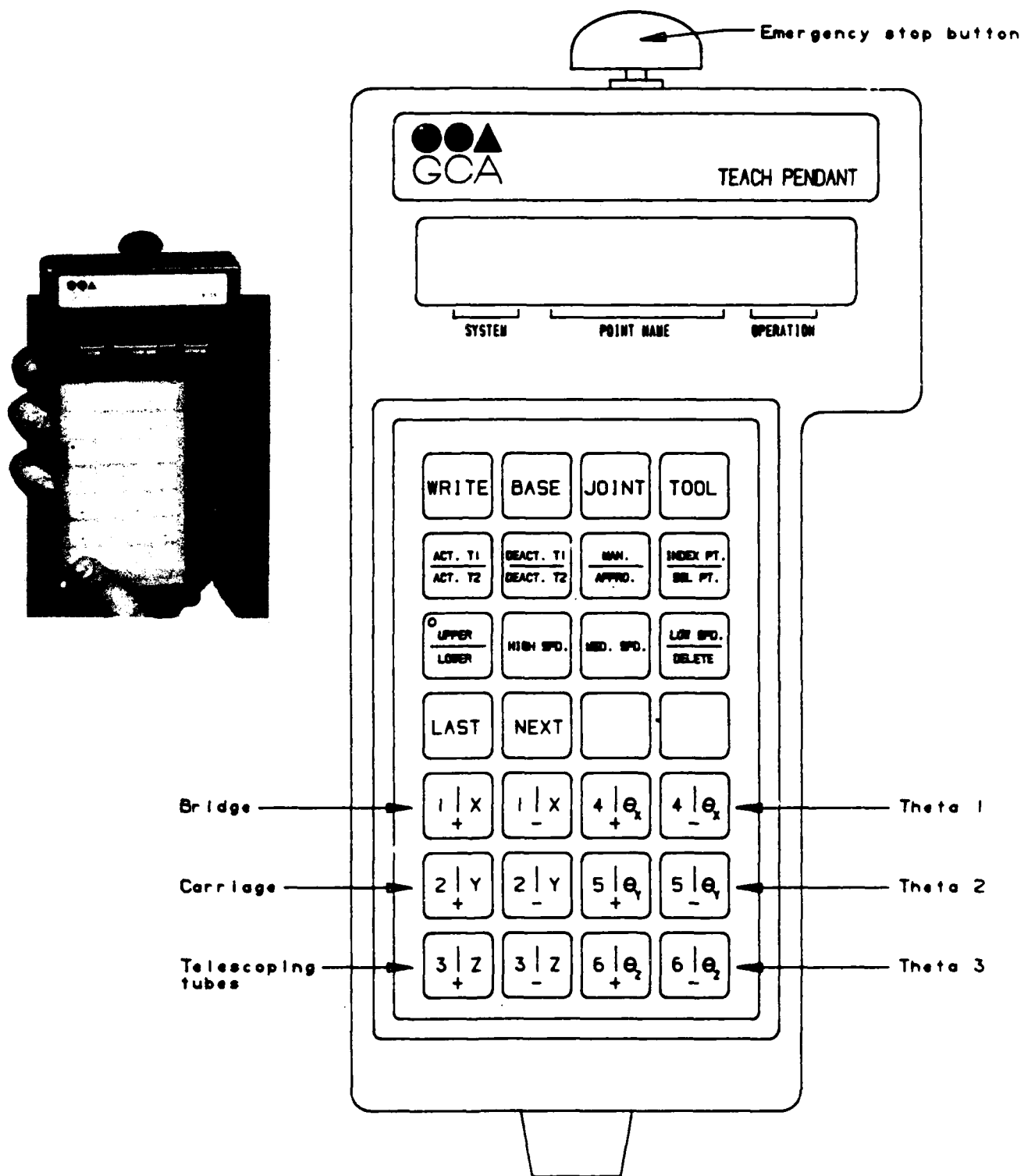


Figure 10. CIMCORP (GCA) teach pendant.
(Courtesy of CIMCORP)

depending on whether the UPPER/LOWER shift key (underneath it) is pressed, lighting an LED. Similarly, the right key deactivates tool 1 (DEACT.T1) and tool 2 (DEACT.T2). Another button selects either simple manual operation or approach operation submodes in the Manual mode, depending on the shift key. In simple manual (MAN), a number of "drive" keys can be pressed at the same time. In approach (APPRO), each joint is tested separately, one key at a time, until it reaches the position it would have for the predefined point; the operator moves the robot in the direction of that taught (specified) point without knowing its actual coordinates. When the joint reaches the point, the pendant display shows "Coordinate OK." If it doesn't, the display shows either nothing or "Wrong Direction" (and the joint won't move). When all six joints have been tested and show "OK," the CRT/keyboard terminal displays a "Point Reached" message. The fourth button in the second row is the INDEX POINT/SINGLE POINT key, which functions with the WRITE key. The UPPER/LOWER shift key invokes the index point operation (for teaching an array of points). The screen shows "Index" in its operation section, and the WRITE key invokes a declaration of a new point, after the prompt "TI." If the UPPER/LOWER shift key is not pressed, the INDEX POINT/SINGLE POINT button invokes a single point operation. The WRITE key defines and advances the current point to the next point. The screen shows "Single." The prompt is "TS."

c. Third row: The first key in the row is, as indicated, the UPPER/LOWER shift key. When it is pressed, a button LED is lighted indicating that the functions designated by the upper labels on dual function keys are enabled. The second button in the row, for HIGH SPEED in the teach or manual mode, gives robot motion on a high speed (which is 50 percent of the maximum speed, in the default state). The third button is for MEDIUM SPEED, which is 10 percent of the maximum in the default state. The fourth button has two functions. In the shifted (upper-label) function, it gives the robot LOW SPEED, whose default state is 1 percent of the maximum. The unshifted function is DELETE. It removes the point name and coordinates of the current displayed point from memory.

d. Fourth row: The LAST button makes the display show the type, name, and index of the previous point, and the NEXT button makes it show these for the next point. With these buttons the operator can skip to any point name by pressing one of the buttons a certain number of times. The third and fourth buttons are unused.

e. Fifth, sixth, and seventh rows: These are the drive keys. The six in the two left columns are numbered 1, 2, and 3 and designated X, Y, and Z (or BRIDGE, CARRIAGE, and TUBE for a rectangular gantry robot). The six in the two right columns are numbered, 4, 5, and 6 and designated theta X (θ_x), theta Y (θ_y), and theta Z (θ_z). Within each pair in each column, the left button is +, the right -; + means joint rotation in a positive direction or motion in a positive direction (along Cartesian axes), whereas - means rotation or motion in a negative direction. The numerals indicate joints, the letters--Cartesian axes or tool angles. Interpretation depends on the coordinate system that has been selected--BASE, JOINT, or TOOL. With BASE coordinates, a selected button among the six on the left moves the tool tip linearly along the selected axis in that system. Among those on the right, the theta X key changes the orientation of the tool with respect to the X-axis and the theta Y key does so with respect to the Y-axis, with the tool tip remaining at the same position in space; the theta Z key rotates the last

joint about an axis associated with the current direction of the tool. With JOINT coordinates, the buttons move the joints separately. With TOOL coordinates, the X, Y, and Z buttons move the tool along the selected axis of the TOOL coordinate system; the theta Z button rotates the tool about the Z-axis of the TOOL coordinate system.

Design Aspects

Coding/Grouping

Push buttons with similar functions are in the same row or columns, but otherwise without spatial separations between groups.

Labeling

Control keys have English words or abbreviations if the word exceeds five letters. Drive keys have letters or numerals.

Feedback

It is not clear whether control keys have LEDs, other than the UPPER/LOWER shift key.

Errors

Apparently error messages do not appear on the pendant, but do appear on the CRT at the CRT/keyboard terminal on the controller panel.

Multiple functions

Five of the fifteen control keys have dual functions, dependent on a shift key; this also serves the drive keys, which also have dual functions.

Miscellaneous

Positive and negative symbols (+ and -) indicate the directions of motion or rotation of the tool center point or wrist, or the individual joints.

CRT/KEYBOARD TERMINAL

The 12-inch, black and white CRT display and slanted keyboard are mounted in the controller panel of the control cabinet, for standup operation, with the CRT at eye level, behind tempered, tinted glass. The keyboard has a QWERTY portion, a numeric keypad, and 10 optionally programmed function keys, 6 on the QWERTY portion, 4 on the keypad portion, numbered S1 to S10. In

addition, a menu system can display options among which a selection is made by typing the option number. The CRT display shows commands and information entered at the keyboard, current operations, and error and status messages, with scrolling.

CONTROLLER PANEL (see Figure 11)

This has the following components:

1. A MAIN POWER switch (ON, OFF) must be off to open the cabinet door.
2. An EMERGENCY STOP push button (palm-operated) cuts power to the servo drives, engages the brakes, and halts robot movement. A key is needed to reset it.
3. A POWER KEYLOCK switch admits power into the cabinet. When on, a green light is lit and only then can the robot function. This power maintains battery backup power, which prevents memory loss.
4. ARM POWER ON-OFF buttons enable the robot to move in the Teach and Manual modes if ON is pressed, and a green light comes on. When the arm power is off and an executed program is completed, a red light comes on. Power is off when the robot is not in use.
5. On the "Execute" panel, a STOP push button turns off arm power and turns on a red light. It initiates a controlled stoppage of operation (and tools) and is used at the end of a shift. The operator restarts with the ARM POWER and RUN buttons. On the same panel, a RUN push button functions in the Execute and Orient modes; it turns a green light on, which goes out if (a) the program is completed, (b) power goes off at the arm, (c) the keyboard initiates a stop, and (d) there's an emergency stop.
6. Four optional user-definable function push buttons indicate a function is being performed by an executing program or initiates a function in an executing program. A light goes on automatically or if a button is pressed.
7. Two speed-selection buttons increase and decrease speed, as indicated by LEDs. They function only during program execution. LEDs also display the percentage of programmed velocity, which is some percentage of the maximum speed.

On top of the controller cabinet are five lights: red for Arm Power on, blue for Brakes Unlocked, white for Run (same function as the Run light on the controller panel), green for Stop (same function as the Stop light on the controller panel), and orange as a spare.

In addition, a sonalert alarm produces an audio warning tone. It is connected to the digital output in the controller.

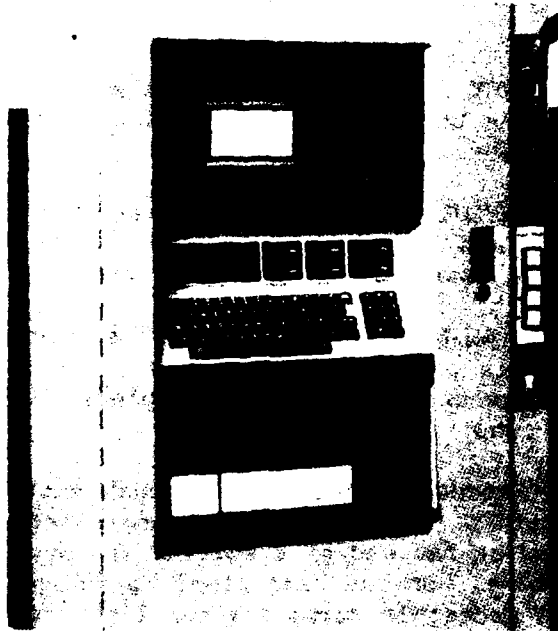


Figure 11. CIMCORP (GCA) controller.
(Courtesy of CIMCORP)

SOFTWARE

There are eight operation modes, selected at the CRT/keyboard terminal: Monitor (M), the base mode from which active modes are accessed; Manual (MN), for maintenance or demonstration with the teach pendant; Teach (TS, TI), which establishes joint coordinates and can be simultaneous with the Manual mode, with the teach pendant jogging the robot; Edit (ED, IN), which writes or changes the program from the CRT/keyboard or a remote terminal and defines paths, operational parameters, and peripheral devices; Orient (OR), which moves the robot to its home position; Execute (EX), for execute or playback; File Handler (F); and Communication.

* Each mode has a separate set of commands, all of which are English words. Most are abbreviated for faster or easier entry, with minima of letters, including some single letters. For example, for "execute program continuously," the command is EXCONT or EXCD; for "execute one cycle only," it's EXCYCLE or EXCY; for "execute one step at a time," it's EXSTEP or EXST; and for "display available programs," it's DIRECTORY or DIR.

Motion control commands include MOVE (the tool tip to a desired location), REMOVE (a relative movement; some incremental distance), PATH (a specified sequence of points), REPATH (relative path), VELOCITY (tool tip speed in execution), AVELOCITY (absolute speed--maximum), ACCELERATE, AACCELERATE, STRAIGHT (path between points), OPTIMAL (optimizes certain moves), CIRCULAR (tool tip follows an arc), ENVELOPE (radius from a point to start a new motion), ACCURACY (radius from a point to execute a non-move command), ORIGIN (new Cartesian origin and axis orientation), ARMSTOP (stops any arm motion, e.g., in a subroutine executed due to an interrupt), ARMRUN (continues or completes a move aborted by ARMSTOP), LOLIMIT (minimal value of a joint coordinate commanded by a motion statement), and HILIMIT. Operands come with most of these. In addition, TMOVE moves the tool tip a specified incremental distance relative to the current tool tip location, in the TOOL coordinate system, and TPATH bears a similar relationship to PATH or RPATH.

Assignment commands, which assign different data types to names, include SET (assigns integer data to variables), POINT (assigns location data to location names), DEVICE (assigns device data to device names), and ANALOG (assigns an analog channel to an analog name). Industrial control commands are INPUT, OUTPUT, WAIT, ENABLE, DISABLE, and DEFINE (assigns a device name or channel number and an interrupt subroutine name to an interrupt level). Other commands include DELAY, DISPLAY, ADISPLAY, BROADCAST, ABROADCAST, ENTER, and TOOL (specifies tool length). Sequential control commands include IF, ELSE, ENDIF, WHILE, ENDWHILE, CALL, RETURN, END, and HALT. There are also program step sequencing commands, communication commands, and 25 edit commands.

Commands come from the teach pendant and the CRT/keyboard terminal. The software includes a menu display subroutine, 241 error and status messages, and numerous prompts. A HELP request displays a list of valid Monitor mode commands. CIMROC[®]2 robot control operates with the CIMPLER[™] language.

HUMAN FACTORS INTERESTS

A company brochure states that "The cabinet of the CIMROC® 2 was human-engineered for ease of operation, maintenance, and security." It cites a CRT at eye level behind tempered and tinted glass, with optional color and programmable block graphics, a "full travel" keyboard "slanted back for optimum programming comfort," a sealed, lockable plexiglass door on the controller cabinet, a front door that swings out for full front servicing, and electronics in modules that swing out for easy servicing.

6. GMF (GENERAL MOTORS FANUC) ROBOTICS CORPORATION

TEACH PENDANT (see Figure 12)

The present teach pendant for the Karel control and software system, described in this section, was preceded by an RC (robot controller) pendant and a number of developmental proposals. The RC pendant has a single-line display, 32 buttons (yellow and white with black labels, and one red one with numerous symbols), 12 lamps at the top, away from the buttons, a mode selector control for three coordinate systems, and six robot motion buttons--three for a positive direction (+) and three for a negative direction (-), used for arm rectangular coordinates and wrist angular coordinates. Not only was it found to be difficult to handle, but a complete revision seemed required. Fanuc (the Japanese partner in GMF with the General Motors Corporation) proposed a pendant with a rectangular array of 38 buttons (with a vertical long axis). Further concepts from General Motors included (a) a rectangular shape, that with a horizontal long axis could be flat and placed on a pedestal or stick, with 6 function buttons and 12 other buttons; it seemed awkward to hold, and the display was too small; (b) a T shape with either 12 movement buttons on the T's vertical segment or a control stick (joystick), plus 6 function buttons and 12 other buttons; (c) a rectangular shape with a horizontal long axis, with a control stick between a numeric pad and an array of 12 function buttons, that would rest on a telescoping pole or a three-legged easel; and (d) another T shape, but with buttons only (no control stick). The buttons in one of the concepts included six soft keys. The method of holding the pendant varied. After considerable consultation, the design adopted by GMF's Product Planning Department was still another approach.

Displays

The teach pendant display can contain 8 lines of 40 characters each. The top line gives the status of the controller, (e.g., Manual); the menu level--Teach; and the coordinate system for jogging--Joint. The next lines identify the position being displayed and its Cartesian coordinates and next position. The bottom line shows the menu options available. There are 13 basic display menus, shown in black on pink or black on light blue, as well as scrolling and a cursor. The Manual menu has as its top level Teach, Testrun, Adjust, User Menu, and KCL (Karel Command Language). At the next level, Teach components are Delete, Append, Insert, Record, KCL, and the Testrun components are Param, Break, Adjust, User Menu, KCL. The Tool menu has at its top level Prev, Next,

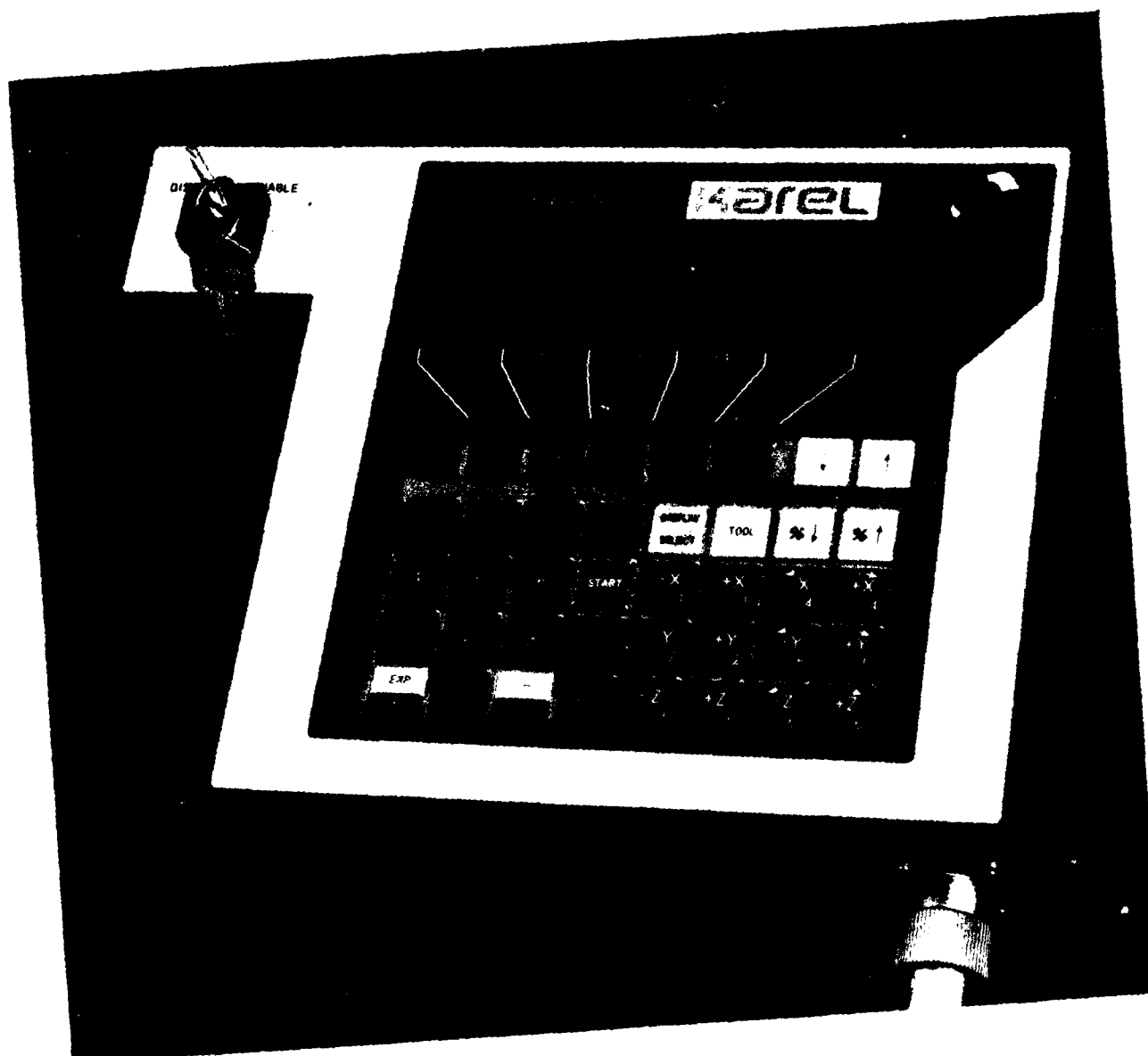


Figure 12. GMF teach pendant.
(Courtesy of GMF Robotics Corporation)

Relax, Open Close; and the DSEL menu has Input, Output, Error, Param, More, and Program. In addition to the menus, the screen displays user and error messages, program variables and parameters, output, and positional data.

Controls

Forty push buttons control 55 commands or functions. In addition, there are deadman switches on the right and left sides, an ENABLE-DISABLE key switch, and an EMERGENCY STOP button. The EMERGENCY STOP button is a big red button at the pendants's top right corner that cuts servo power, applies brakes, and aborts the program. The ENABLE-DISABLE switch, with a key to protect it, enables or disables the teach pendant. When it is disabled, the operator can still monitor system and program variables and input and output plus receive messages. The EMERGENCY STOP and the HOLD buttons remain enabled; in the Enable position, robot motion commands can be executed.

a. Below the screen are five large, soft buttons that take on the functions of the menu items displayed above them.

b. A PREV MENU key to the left of those buttons puts the previous menu (level) on the screen. Two cursor keys (labeled black on pink) to the right of the buttons scroll the screen; also, with the SHIFT key, they can skip array elements to the next variable.

c. A numeric keypad (with white on black numerals) includes a dual-function Exponent/Decimal (EXP/.) key; the decimal point portion is white on black, and the exponent value half is black on white.

d. The SHIFT key that enables the upper functions of the dual-function keys is colored white on blue.

e. An ENTER key enters data inputs from the teach pendant.

f. A START key (green) begins program execution that continues while it is held down with the SHIFT key at the same time.

g. A CANCEL key (white on black) cancels teach pendant data input. This is a dual-purpose key. The other half is labeled by a minus sign (black on white) and produces negative values.

h. Two speed override keys increase or decrease (shown by arrows) displayed current speed by 5 percent with each press.

i. HOLD (red) on a dual-purpose STEP/HOLD button decelerates and stops the robot; it is released by pressing it again. STEP (black) executes a program in a single-step mode.

j. A DISPLAY/SELECT key is color-coded with black lettering on a buff background, and next to it a TOOL key is coded with black lettering on light blue.

k. There are 12 paired Manual Jog buttons (white or dark blue) in a 4 x 3 array. Those on the left move the robot in a negative (shown by a minus

sign) direction, those on the right in a positive (plus sign) direction. The left three pairs move the arm (tool center point) along coordinates or its joints individually; the right three buttons control movements of the wrist along coordinates (or its joints). The buttons are dual-purpose; the SHIFT key enables the coordinate movements. These are labeled X, Y, and Z, and the buttons for the wrist coordinates have curved arrows as well. The labels for individual joints are numerals.

Design Aspects

Coding/Grouping

Color-coding of push buttons is used extensively, even for the two halves of the dual-purpose push buttons. (An earlier pendant design also used color-coding, with the motion keys in white on light blue and the keys above them in yellow.) The soft function buttons are configured and labeled so their association with the assigned functions above them on the display is definite.

Labeling

Words are spelled out. Arrows are used as symbols.

Feedback

Results of keyboard actions are shown on the display.

Errors

The screen shows error messages.

Multiple functions

Four buttons have dual functions, as do the 12 robot motion keys. The five soft function keys also have multiple functions.

Miscellaneous

A slot between the pendant frame and a vertical handle lets the pendant be grasped directly by the handle or possibly with the hand through the slot.

CRT/KEYBOARD TERMINAL

The CRT and keyboard are mounted in a recess on the controller panel, for standup use. The keyboard is angled at 40°; the keys are the membrane type with a QWERTY arrangement. (The predecessor keyboard for the robot controller had only the NC letter keys and lacked soft buttons.) Keys include \uparrow for BACKSPACE, DELETE, CANCEL, RESET, and CURSOR/SCROLLING; some are colored red. Just below the CRT are five unlabeled soft function keys for the menu system. On their left is a PREVIOUS MENU key, and on their right a Karel/KCL function that alternates the display between Karel and KCL screens. All commands can be typed in or entered by the soft function keys. The soft keys and their labels on the screen prompt the user, who enters the command at the KCL prompt (a >).

As in the teach pendant display, the bottom line on the CRT identifies the varying function keys below it. At the top, the display (in the power-up stage) first identifies the software version, unit model, any error message, and the program line number. Then it shows the status of the current Karel program, followed by error history--the last 10 errors since power-up. Next it displays the position of the robot, in joint, world, or user coordinates. A Karel screen displays outputs of the currently running program or requests information from the user. A KCL screen displays command prompts and identifies the KCL soft function keys. There are seven groups of KCL screens at the initial menu level. Further menu levels vary in number. Program Control has two screens, Data Manipulation--three, Program Development--two, File Maintenance--two, and Miscellaneous--two. Other groups are Diagnostics, Utility, and Communication. The KCL screen has a prompt/error line to indicate errors in command inputs and prompts for more information. Forty error messages are available for indicating errors in KCL command entries. These include semantic errors and syntax errors (spelling and punctuation), such as unrecognized key words, invalid constants, and too many arguments.

CONTROLLER PANEL (Operator Panel) (see Figure 13)

On the right side of the panel in descending order are the following controls:

1. A green POWER ON push button with a red light above it.
2. A red POWER OFF push button.
3. A REMOTE OFF/ON selection switch - When off, all panel controls are enabled. When on, the only components enabled are Power On, Power Off, Hold, Emergency Stop, Fault Reset, Overtravel, Overtravel Release, and lamps.
4. A MEMORY PROTECT ON/OFF key switch - When on, Karel commands can be executed. When off, programs and data can be put into secondary memory storage.
5. A large red EMERGENCY STOP push button - It cuts off servo power, applies brakes, and aborts the program. Above it is an Hour Meter display showing the accumulated time that the robot is on.

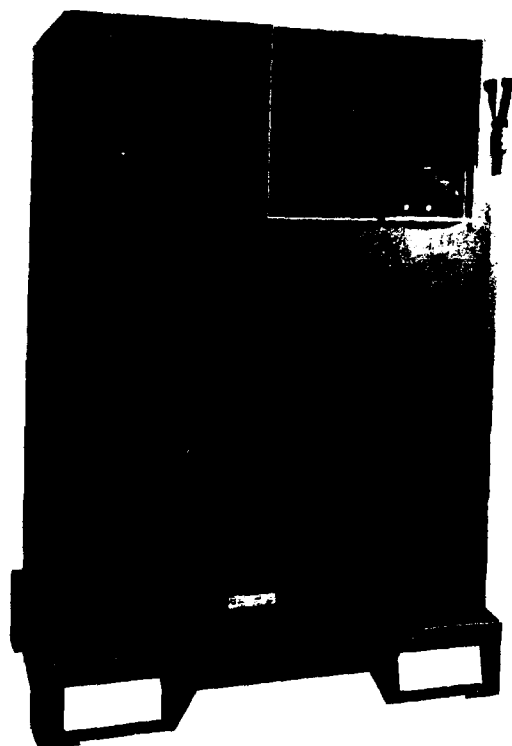


Figure 13. GMF controller.
(Courtesy of GMF Robotics Corporation)

On the left side in descending order are the following:

1. Three lamps in a row - (a) A green Panel Enabled lamp, when on, shows that the panel has been enabled; the Remote switch has been set to OFF. (b) A green System Ready lamp, when on, shows that the servo system has power and the robot can move. (c) A green Teach Pendant Enabled lamp, when on, shows that the teach pendant has been enabled and has motion control priority.

2. Two lamps in a row - (a) A red Robot in Cycle lamp, when on, shows that the robot is running. (b) A red Not Calibrated lamp, when on, shows that axis calibration has not been performed.

3. Two push buttons in a row - (a) A green Cycle Start button, located below the Robot in Cycle lamp, executes a program or resumes it. The panel must be enabled and the software at \$CYCL STRT. (b) A white Calibrate button, located below the Not Calibrated lamp, executes axis calibration procedures.

4. Two lamps in a row - (a) A red Cycle Stop lamp, when on, shows that the Cycle Stop button, below it, has been pressed. (b) A red Robot Held lamp, when on, shows that the Hold button, below it, has been pressed.

5. Two push buttons in a row - (a) A red Cycle Stop button stops program motion at the end of the current cycle. It can be the same as the Hold button, depending on the software. (b) A red Hold button stops program execution, decelerates, and stops the robot. This is the normal procedure for halting the robot.

6. A red Robot Fault lamp indicates there is a fault in the system. Under it is a red Fault Reset push button that clears an alarm condition. If there has been no alarm, it makes the program pause and holds robot motion.

7. A white Home push button moves the robot to a position specified by the software as \$HOME. Under it a white Overtravel Release push button applies power so the robot can be jogged out of overtravel.

8. Two ports, lamps, and connect/disconnect switches serve the RS/232C connector and the teach pendant.

(The predecessor panel for the robot controller cabinet had only six switches, including the EMERGENCY STOP and a selector switch. The new cabinet has an 8-inch space under its bottom, to enable forklift operations and to raise the height of the CRT and the keyboard.)

SOFTWARE

In 1984 the SmartWare workstation allowed programs to be written in the S or G codes of CNC (computer numerical control) or in a robot programming language, which uses equivalents in English-like mnemonics. In 1985 the SmartWare off-line robot programming system provided a method of generating programs for Karel controllers. A company brochure referred to "Programming

in powerful user-friendly programming/editing language that uses descriptive English words or traditional Karel language codes....Menus provide step-by-step instruction to the inexperienced programmer. Those menus can be bypassed if desired by the experienced programmer." Jacobs (1984) noted that about 80 percent of the program statements in modern robot programs are logic statements, rather than positioning commands. In 1985 Karel was introduced as a complete, integrated system of software, hardware, and programming tools; further references to SmartWare dropped out.

The Karel language is described as a high-level programming language that provides functions like those of conventional computer languages. A Karel program is a list of instructions in special English-like statements, including 88 reserved words, user-created identifiers, 30 operators and special characters, and 134 user-accessible system variables. The Karel Command Language (KCL) is a set of commands that are used to communicate with the controller. As noted in the sections on the teach pendant and CRT/keyboard terminal, users communicate commands through the pendant and terminal and a menu system (or directly) to manipulate data and files, monitor system status, enter and run programs, change system variables, initiate calibration, and remotely modify, debug, and run programs. There are five categories of error codes: WARN (nonfatal), 477 items; PAUS (recoverable), 27; ABRT (aborts an executing program), 128; CNCL (cancel current motion), 4; FATL (no recovery--must restart system), 97; and DBUG, 12.

HUMAN FACTORS INTERESTS

Anthropometric improvements in the controller cabinet (e.g., raising the height of the CRT, tilting the keyboard) are of interest to GMF along with the evolution of the teach pendant. Concerning the evolution of the teach pendant, GMF reviewed "the ergonmetric [sic] issues of the design proposals" with a local design company, which produced artist sketches, and discussed concepts with Fanuc. GMF's Product Planning Department reports it also surveyed key automotive and nonautomotive customers. Jacobs (1984, pg. 4-5) wrote concerning the workstation, "During development, we tested the system's ease of use as follows. We took several people from GMF and several outside customers, sat them at the workstation, instructed them in the use of the system for 5 minutes and stood back and watched their programming session. This not only proved our success but also pointed out several areas that required improvement. These improvements were incorporated into the final system release." Jacobs added: "The workstation was tested at several beta test sites. These sites are GMF robot customers who were interested in purchasing this workstation. They were given a workstation on consignment and used it to program their robots. They discovered a couple of 'bugs' that we did not find, but, more importantly, made some product improvement suggestions that we incorporated into our final system release which many of our users are finding helpful." (1984, p. 4-7)

7. INTERNATIONAL BUSINESS MACHINES (IBM) CORPORATION

TEACH PENDANT (Pendant) (see Figure 14)

IBM uses a teach pendant for its 7565 robot, which in its maximum configuration has six axes (degrees of freedom) plus a gripper, and as an option for its new 7575 and 7576 four-axis robots. (Other 7565 robots have three or four axes.) The three arm axes have linear (translational) motion, in a box-like frame of reference; joints are designated JX, JY, and JZ for X, Y, and Z travel. The three wrist axes have rotational motion; the joints are designated JP for pitch travel, JW for yaw travel, and JR for roll travel. The gripper is designated JG (for gripper travel). Other IBM robots (Scara type) are controlled through the CRT/keyboard personal computer terminal (similar to that with the 7565). The terminal can be situated close to the robot so the programmer can observe the tool tip adequately from the terminal (and thus does not need to carry a device to observation locations) and can teach the robot by moving it with direct manual manipulations (and thus does not need a device to move it) or by commands at the terminal. The 7565 pendant has a 12-character display, 8 LEDs, and 32 keys in a 4 x 8 matrix, numbered 101 through 132. (The 7575/7576 pendant has a 4-line, 64-character LCD and 32 LEDs.) Eighteen of the keys are program function buttons, but only one of these is predefined, the Attention key. Subroutines are assigned to the other keys and can be invoked by them. They are used for many purposes, for example, to start and stop an operation, to start hydraulics and enter a Run mode, for manual control of user devices, and for manual control of recalibration routines. Below the function keys are 14 robot motion keys, six for the arm movements (X, Y, and Z), six for the wrist movements (P, W, and R), and two for the gripper. For each arm movement there is a + direction key and a - direction key, and for each wrist movement there is a CW (clockwise) key and a CCW (counterclockwise) direction key. One key opens the gripper; the other closes it.

CRT/KEYBOARD TERMINAL (Display Terminal)

This VDT has the familiar QWERTY layout as well as keys for inserting and deleting characters and lines and four motion keys, and it has a numeric keypad. Eight of the keys in that keypad have dual functions. Holding down an Alt key changes the function from a number to a program function. These have default values the user can replace--six of them editing functions, such as going forward or backward 10 or 21 lines or editing a last line or a list. One key retrieves the last input and prefills the next read with it. An Attention function terminates a current motion or breakpoints from an executing program and displays a prompt at a higher subroutine level.

CONTROLLER PANEL (Manipulator Operator Control Panel) (see Figure 14)

This panel is situated on top of the horizontal element of the frame that supports the robot and is backlit. In addition to an Emergency Hydraulics Off control, it has five keys. Two are for hydraulics. The others are Run

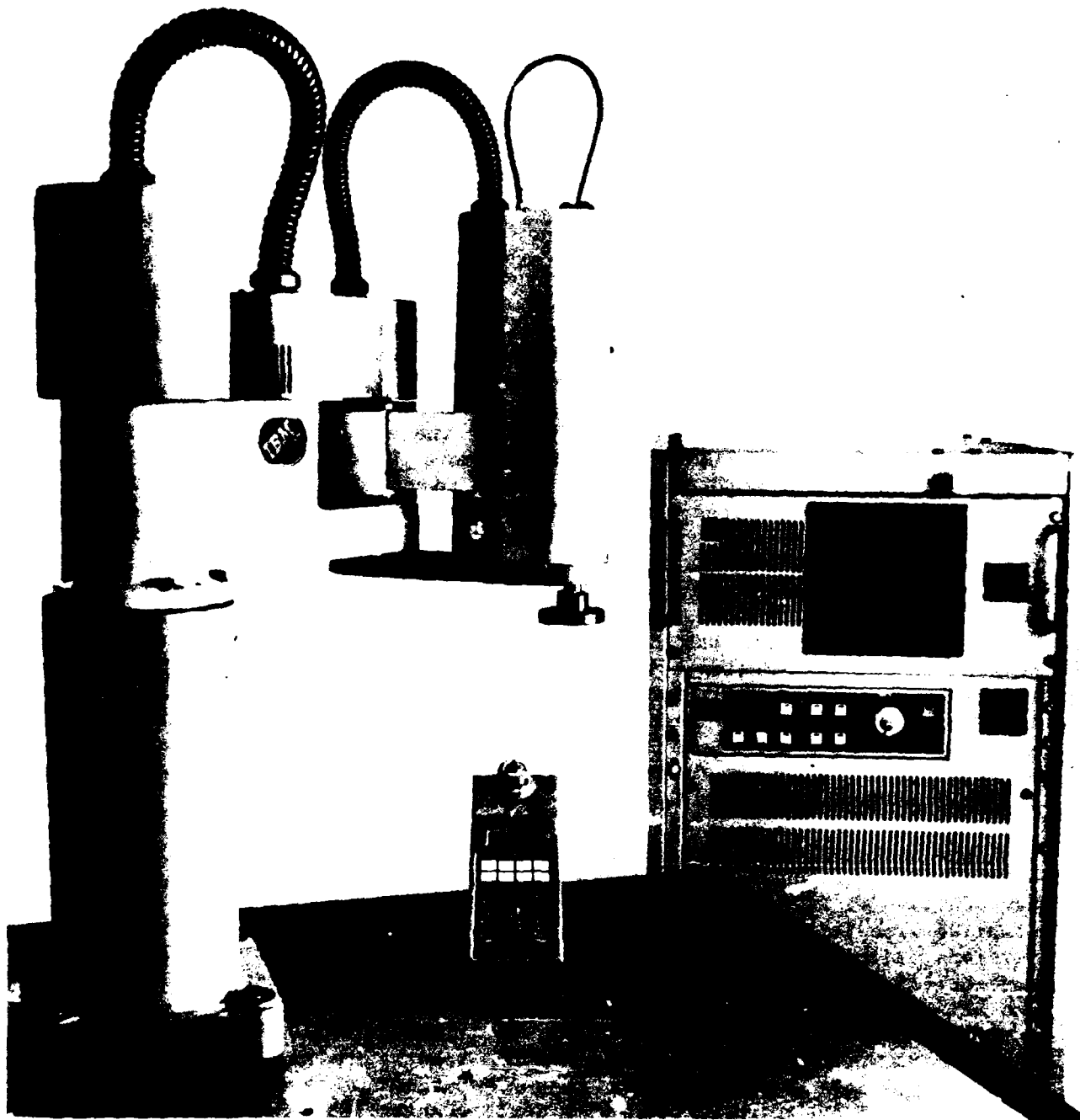


Figure 14. IBM teach pendant and controller panel.
(Courtesy of International Business Machines Corporation)

Motion, Initiate Freeze, and Interrupt Request. In the Idle mode, the Freeze and Run keys are inactive. In the Freeze mode, the Run Motion key is active. If pressed, it puts the system into the Run mode with the subroutine assigned to Run Motion. In the Run mode the Freeze key is active. If it is pressed, the subroutine assigned to Initiate Freeze is executed. The robot changes to Freeze immediately if its speed is less than 0.5, otherwise at the end of the motion. Various conditions and constraints occur with these keys/functions. For example, if a motion executing when Initiate Freeze is pressed for 5 seconds, the system turns off the hydraulics and enters the Idle mode. When Freeze is initiated, a Freeze Motion light flashes. Hydraulics are turned off so the operator can enter the area. Essentially, the robot proceeds from Idle to Startup to Freeze to Run.

SOFTWARE

As might be expected for smaller robots primarily for assembly, IBM has placed more interface emphasis on software than on hardware, and on factory-floor programming through the CRT/keyboard terminal with the aid of a menu system and 178 subroutines invoked there. For example, typing or selecting the command GUIDE lights an LED and enables the pendant to move a set of joints; the FINDPOST subroutine is used in calibration; the MOVE subroutine can include speed elements; the DMOVE subroutine enables a move relative to a current position, and AMOVE, enables a move before a previous move is completed, while WAITMOVE waits for AMOVE to complete, and STOPMOVE stops motion already in progress; the SPEED system subroutine selects robot speed (as a fraction of full speed), along with ACCEL, DECEL, and SETTLE subroutines. The QGOAL subroutine returns the last joint destination issued, QPOSITION the commanded position less servo system feedback.

The Screen Editor has three types of operations: (1) Obtaining the editing object information (five subroutines); (2) changing the current line (two subroutines); and (3) modifying the object information (six subroutines, i.e., delete, renumber, insert, change, copy, and move). Subroutines DETRACE and TRACE are used to locate problems. The Screen Editor is used to create or alter files on disks or diskettes and AML (a machine language) programs or application data sets and to edit AML subroutines in storage.

AML is described as a base language for functional enhancement by writing subroutines and applications programs. AML-Entry has been a simplified version of AML, first used in 1982 and 1983 for the 7535 robot controller compiled with IBM's personal computer, and proceeding through a series of four versions, with subversions, by 1984, for the 7547 and 7540 as well. It is menu-driven with English-like commands. AML was first marketed in 1982. AML and AML-Entry (AML-E) are becoming fairly similar, with the same syntax and subroutine structure. The 7565 robot is controlled by the IBM Series II/1 computer with AML. (The 7575 and 7576 robots use the second-generation AML/2.) It has been estimated that 90 percent of applications programs generated with AML or AML-E deal with logic changes, equipment interfaces, and other factors, rather than robot movements.

Another software development has been the development, for the 7565 robot, of PRBE--Programming Robots by Example. PRBE creates a program, runs

it, and updates it. Then it can be converted to AML for additional programming and use of the AML subroutines, systems subroutines, and identifiers. PRBE operates primarily through menus of commands rather than typing command terms. The CRT/keyboard terminal's screen presents options with numbers, and the user enters the appropriate number through the soft keys. English-like commands are explained on the screen, instructions and prompts are provided to minimize recourse to reference manuals, and error messages (42 error and information messages) are displayed concerning incorrect data. Commands with operands constitute program statements, leading to action when the program runs, and they define points or arrays for an immediate response. A secondary screen describes the function selected by a command and calls for an operand entry. The user can also "type ahead" to bypass succeeding screens by entering commands and operands directly. The user is urged to keep notes on worksheets to help in writing program statements later. The three sets of screens (1) define points, with such commands as Approach, Grasp, Orient, Position, Reach, Set Gripper, Transport, Withdraw, Jog Joint, Jog X, and Pendant; (2) define arrays; and (3) invoke subroutines as program statements with such commands as Delay, Expect, If Off, If On, Index, Set Speed, Set Off, and Set On.

After identifying a program, the user calibrates the robot by selecting a Calibrate Base Point from a Program Options menu, then proceeds through two more menus and three prompts, moving the robot with the pendant or the Findpoint command. Thereupon the user selects Define Points Relative to a Base Point from the Program Options menu and proceeds through two more menus and two prompts. At this point the user moves the robot with the pendant or the keyboard to a position, and on the keyboard enters a command (the action to take) and an operand (distance to move, pressure, point name). The user is advised to go between the keyboard and the pendant "as frequently as you need to." The user presses the ENTER key at the keyboard to save the location of the point in the computer, verifying it by entering information from the keyboard or imitating the sequence by using the pendant. Thus, keeping notes, the user defines all points (robot positions) needed before entering program statements. (Previously defined points may be referenced.) Hereupon the user using the worksheet begins to write the program statements (entering them into the computer), each consisting of one "step," incorporating the defined points; the user is prompted for statement numbers and helped by a menu listing 19 usable command options. PRBE does not produce a full applications program. As noted earlier, it must be converted to AML and developed further with the AML subroutines.

HUMAN FACTORS INTERESTS

Although a human factors course has been given for IBM programmers and engineers involved in robotics, publications have not pinpointed the term. However, consideration has been given to (a) feedback from users and (b) user skills. "When it comes to market feedback," according to Grossman and Short (1985), "technology leaders have to expect the unexpected. And the opinions on AML spanned the spectrum. Perhaps the most favorable feedback came in the Japanese press, which cited AML as being the most advanced commercially available robotic language. On the other hand, some of the users identified important deficiencies. In response to this inconsistent feedback, we made a

number of modifications to the software. In the process, we focused on AML's role as a language for the IBM 7565 robot." In addition, it is understood that new commands and other improvements in AML-E resulted from user requests, both inside and outside IBM.

P. B. Van Dyke (personal communication, August 1985) has suggested a "pyramid" concept of skills and software applicable to robotic developments. System programmers and operating systems form the base. At the next higher level are highly skilled applications programmers and AML with its subroutines. Up one level still are somewhat less skilled applications programmers and subroutine packages. Close to the top are applications programmers or manufacturing engineers using PRBE or RAPID (Robot Applications Programming by Interactive Display), and finally actual applications requiring the greatest ease of use. Another pyramid concept has three interface levels: (1) applications programmer; (2) pendant user (shift supervisor or repair technician); and (3) controller operator (concerned with power supply, calibration, application selection, start/stop, and errors).

8. SCHRADER BELLOWS DIVISION OF PARKER HANNIFIN CORPORATION

The two robots manufactured by this company, MotionMate and MotionMate II, are pneumatically powered, four-degrees-of-freedom (and gripper) robots with base rotation, translational lift, translational extension, and wrist rotation (roll). The extent of movement of each segment is controlled by mechanical stops that are adjusted and preset for distance for each task. Speed is also preset. The robot is taught with a teach pendant only by moving individual axes, not by moving the entire arm (tool center point) along Cartesian coordinates. Thus, control of these robots represents what was typical prior to the advent of servo feedback control and programming by teaching motions along rectangular coordinates. There are two teach modes. In one (A-programming in real time), the segments in execution do just what was done in teaching the sequence, including the lengths of pauses between steps. In the other (B), the programmer makes two passes through a sequence. In the first step, the programmer presses the STEP key repeatedly for the sequence, which the robot learns from the output signals--but it does not learn any time data. In the second pass, the time the operator takes between steps becomes the programmed interval. The second mode can correct the timing and input data from the first mode.

TEACH PENDANT (Command Module) (see Figure 15)

Displays

There are no displays except for seven LEDs, six with buttons and one Enable LED that lights 5 seconds after initializing. All LEDs flash four times when an error occurs.



Figure 15. Schrader Bellows teach pendant.
(Courtesy of Schrader Bellows Division of Parker Hannifin Corporation)

Controls

Twenty-four push buttons are situated in two arrays, plus a keylock on the right side that locks out the top row of command keys, thus eliminating any activation of the teach functions during normal operation.

a. Two rows of buttons in an upper array for Command

- Row 1
 - (a) TEACH MODE A (with LED)
 - (b) TEACH MODE B (with LED)
 - (c) END TEACH MODE (with LED)
 - (d) STEP (starts timing the duration of a motion; another press ends this motion and times the next)
- Row 2
 - (a) RUN (with LED) (starts sequence that continues till PAUSE, Power Off, or Reset at controller panel)
 - (b) SINGLE CYCLE (with LED)
 - (c) PAUSE (with LED) (in Mode A, stops the robot)
 - (d) INIT (initialize--returns robot to the start position in the program; then the operator restarts with RUN)

b. Four rows and four columns of buttons in a lower array for Control

- (1) Six AUX (auxiliary) keys to send output signals via output terminals on the controller back panel
- (2) Ten robot movement keys
 - (a) Two for the gripper, labeled GRASP and RELEASE, not adjoining, in different rows and columns
 - (b) Two for base rotation, labeled by horizontal arrows in opposite directions, not adjoining, in same row but different columns
 - (c) Two for arm extend/retract (linear), labeled by vertical arrows in opposite directions, not adjoining, in same column but different rows
 - (d) Two for arm lift and lower (linear), labeled by inverted Ts with arrowheads pointing up and down, not adjoining, in same row but different columns
 - (e) Two for wrist roll, circles with arrowheads, pointing clockwise and counterclockwise

Design Aspects

Coding/Grouping

Command buttons are an array spatially separated from the other buttons. No two buttons for movement control for the same function are together.

Labeling

Most words are spelled out, except INIT. Graphics with arrows indicate directions of motion but not the joints involved except these can be inferred from the directions and curvature or linearity of motion.

Feedback

Any key press produces a beep sound simply to show there was a press. In addition, six buttons light LEDs.

Errors

All LEDs flash in reaction to an error, but only four times.

Multiple functions

No key has more than one function.

Miscellaneous

The arrow base and extension movement buttons are placed for symmetry around one of the gripper (GRASP) buttons (a filled circle) rather than in functional adjacency. In addition, as a result, the RELEASE button is separated from the GRASP button.

CRT/KEYBOARD TERMINAL

None is provided or needed.

CONTROLLER PANEL (Robot Commander)

1. An OFF/ON switch controls power to the controller and pendant.
2. A RUN key has the same function as on the pendant.
3. An INIT (initialize) key has the same function as on the pendant.
4. A RESET key (a) restarts the system, and (b) turns off all outputs.

5. An Output display panel contains 16 LEDs, 10 for robot movement and 6 for auxiliary control.
6. An Input display panel contains eight LEDs.

The panel also has a connector for the pendant.

HUMAN FACTORS INTERESTS

A brochure states that the command module (teach pendant) "is easy to operate. It uses simple graphic symbols for robot commands. LEDs indicate the mode being used, and an audible tone is emitted on key depression."

9. TOSHIBA CORPORATION, TOKYO, JAPAN, and HOUSTON, TEXAS

The units described here serve the System Robot Series (SR-654H 4-axis robot and SR-606V 6-axis robot), Assembly Robots (SR-414H 3-4 axis robot), and the Material Handling Robots (SR-2006V, SR-1806V, and SR-2206V 6-axis robots). All have articulated arms.

TEACH PENDANT (see Figure 16)

Displays

A small LED displays three characters for POINT NO. An LED is associated with every push button except the RESET and robot motion buttons. Three LEDs at the top left light to indicate ALARM (if an error occurs), OVERTRAVEL/HARD and OVERTRAVEL/SOFT (in case of overtravel).

Controls

- (a) A red EMERGENCY STOP button is at the top right.
- (b) Under it is a RESET button that resets the CPU in the pendant.
- (c) Under the display are two POINT SHIFT buttons labeled BACK and FWD, for selecting the point number in the display.
- (d) Beneath these are three TEACH SPEED buttons labeled LOW, MED, and HIGH for selecting the speed for robot arm movement.
- (e) In the next lower row are two COORDINATE buttons, JOINT and WORLD, for selecting the type of robot movement in teaching it, by individual joints or along Cartesian coordinates.
- (f) In a bottom row are three buttons labeled ARM/FREE (for setting or resetting an arm free mode, in which the segments can be moved by hand); BACK

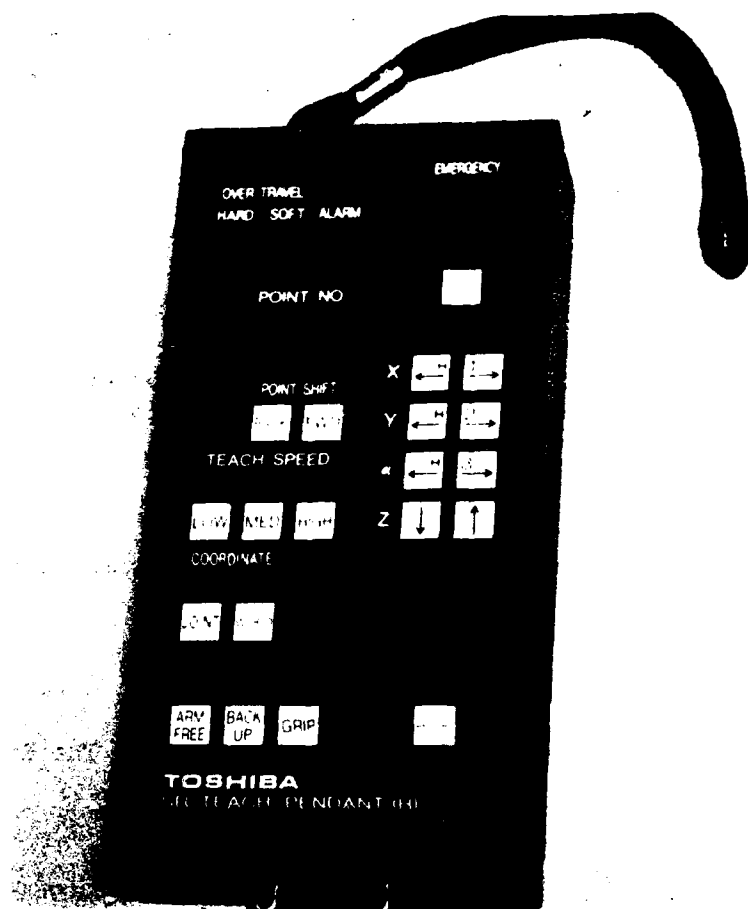


Figure 16. Toshiba teach pendant.
(Courtesy of Toshiba Corporation)

UP for turning on and off the backup signal; and GRIP, for opening and closing the gripper (the LED lights when it is closed).

(g) At the bottom right is a RECORD button to record arm positions in the Teach mode; its LED lights for one-half second.

(h) In an array of two columns on the right of the pendant face are eight robot motion buttons. Those in the left column move the robot to the left or down (indicated by arrows), and those on the right to the right or up (also indicated by arrows). For the four-axis robots, labels to the left of the array identify them as X, Y, α , and Z, and the buttons also have numbers. Pendants for 6-axis robots have six buttons in each column instead of four. The function of each button is controlled by the buttons already mentioned labeled JOINT and WORLD.

Design Aspects

Coding/Grouping

The various sets of buttons are well-separated from each other spatially but are not differentiated by color except for the EMERGENCY button.

Labeling

Words of more than five letters are abbreviated on the buttons, but not on the labels OVERTRAVEL and COORDINATE above them.

Feedback

LEDs indicate when a button has been pressed. The RECORD LED lights for only one-half second.

Errors

An error lights an ALARM LED.

Multiple functions

There is no shift button. Only the robot motion buttons have two functions, selected by the JOINT and WORLD buttons.

Miscellaneous

A strap on the top of the pendant can be used to carry it or to put around the wrist to prevent accidental dropping.

CRT/KEYBOARD TERMINAL (TSR Program Loader; see Figure 17)

This is a portable unit with a handle at the top. It has a liquid crystal display (LCD) with 2 lines of 20 characters each. It has an EMERGENCY STOP button, a RESET button, and six sets of push buttons, with a label for each set. On the right are 20 DATA KEY buttons, with numbers, punctuation symbols, and 10 letters (only). To its left is a PROGRAM CONTROL array of 20 buttons with the following labels: IF, IFS, THEN, SIG, GOTO, GOS, RET, PUL, FOR, STEP TO, NEXT, DLY, BY-SFT, REM, SET SETI, PAU, STOP, PNT, and SPC; one button is not used. These are instruction keys for flow control of program instructions in editing SCOL-1 programs. To their left is an array of 15 ARM MOTION buttons (with one blank): MVP, MOV, MVS, MVC, RDY, ALWY, CRS, FINE, GAIN, UP, DOWN, SPD, OPNI OPN, and CLSI CLS. These are instruction keys for arm motion and grip operation in editing SCOL-1 programs. On the left side of the pendant face, at the top, are four OPTIONAL CONTROL buttons, numbered 1 through 4, with LEDs, for selecting option modes in position definition. Below them are four OPERATIONAL CONDITION buttons with LEDs, labeled READY POS, SOFT LIMIT, TEACH MODE (display mode), and DATA MODE (coordinate mode). In a row below these are four more OPERATIONAL CONDITION buttons with LEDs: OFF SET, ZERO SHIFT, FILE TRAN (for transfer designation), and MEMO MODE (for memory mode). In the bottom row of EDIT MODE buttons are EDIT, INS, DEL, and ERA.

CONTROLLER PANEL

Though the available description is not entirely clear, this panel appears to have 24 switches, including a large red Emergency button at the top, right; a two-position Central Power keylock switch; another multi-position Central Power switch; a Power push button; and controls or lamps for Computer Control, Remote Control, Peripheral Devices Control, Hand Control, Program Control, Operation, Velocity Setting, Hand Attitude Control, Hand Direction Control, Circular Interpolation, Linear Interpolation, and Positioning.

SOFTWARE

Toshiba robots operate with the "high-class robot language SCOL," which stands for "Symbolic Code Language for robot," described in a brochure as "extremely proximate to natural language, allows anyone to easily apply programming," with "Command groups--including arm motion control, program control, and path control." Examples (given) of SCOL (movement) commands are (a) MVS PNT1--PTP transfer up to Point 1; (b) MVS PNT1--Linear transfer up to Point 1; and (c) MVC PNT1, PNT2--Circular arc transfer through Point 1 and 2. Examples given of SCOL program control commands are (a) IF--condition judgment; (b) GOS--subroutine control; and (c) RET--subroutine return. SCOL displays errors on the teach pendant and CRT/keyboard terminal (TSR program loader with an LCD) coded as ER with a numeral or letter. The 45 errors include 15 editor operation errors, 11 editor syntax errors, and 19 SCOL-1 errors.



Figure 17. Toshiba program loader.
(Courtesy of Toshiba Corporation)

HUMAN FACTORS INTERESTS

According to a brochure, "Program teaching is greatly facilitated by pendant teaching and the high-class robot language SCOL." Though detailed information about SCOL was not available, of some interest is the fact that only the letters, P, I, J, K, L, M, N, S, T, and U are available on the DATA Key keyboard of the portable CRT/keyboard terminal--in that order, from top to bottom; an O appears at the bottom, but it is unclear whether it is a letter or a zero, since it follows the typeface and keyface position of neither above.

10. UNIMATION, INC. (WESTINGHOUSE)

TEACH PENDANT (see Figure 18)

The teach pendant described here is for the electrically powered PUMA series of robots that have up to six degrees of freedom--except for the PUMA 100 (with 3-1/2 DOF) whose pendant has 16 push buttons, and for the Unimate hydraulically powered robots (Unimate 2000 and Unimate 4000 with 3 to 6 DOF), and the Unimate (gantry) 6000 with 3 to 5 DOF.

Displays

An LED display has a single line of characters; there are also six indicator lights above associated push buttons and four other indicator lights. These four lights indicate (a) that the selector switch on the controller is in the HALT position; HOLD has been selected on the teach pendant, or an internal hold is activated; (b) the RUN button has been pressed; and (c) the robot has not been calibrated. The LED display can show 23 messages, in addition to "Unimation": six Limit Stop messages numbered 1 through 6 (indicating that the specified joint has reached its software stop); Manual Mode (the robot arm can be controlled by the teach pendant); Comp Mode (the robot arm is under the control of the computer/controller); Prog Running (a user-written VAL program is being executed); No Points (indicating blank storage locations in a specific memory); Teach Mode J (the T command has put the system in the Teach Mode for joint-interpolated motion); Teach Mode S (the TS command has put the system in the Teach Mode for straight-line motion); Error-No VAL (an initialization query has not been answered in 6 seconds, or VAL is not active); Fatal Error (caused by a hardware failure, and changing to Error-No VAL after 6 seconds); Arm Pwr Off (arm power has not been applied); Brk Rls On (the brake release on the controller has been turned on); 40V Bad (+40V and -40V are not equal within 8V for joints 4-6); JT1-2-3 Bad (the Pwm amplifier has a fault); JT4-5-6 Bad (overcurrent has been detected in the motor or power transistor in joints 4, 5, or 6); JT7 Bad (overcurrent has been detected in the end effector or its transistor); HOLD (the RESTART/HALT/RUN switch on the controller front panel is in the HALT position); DBC Off (the dynamic braking contractor is off); and 1-2-3 Limit (a limit switch has been activated).

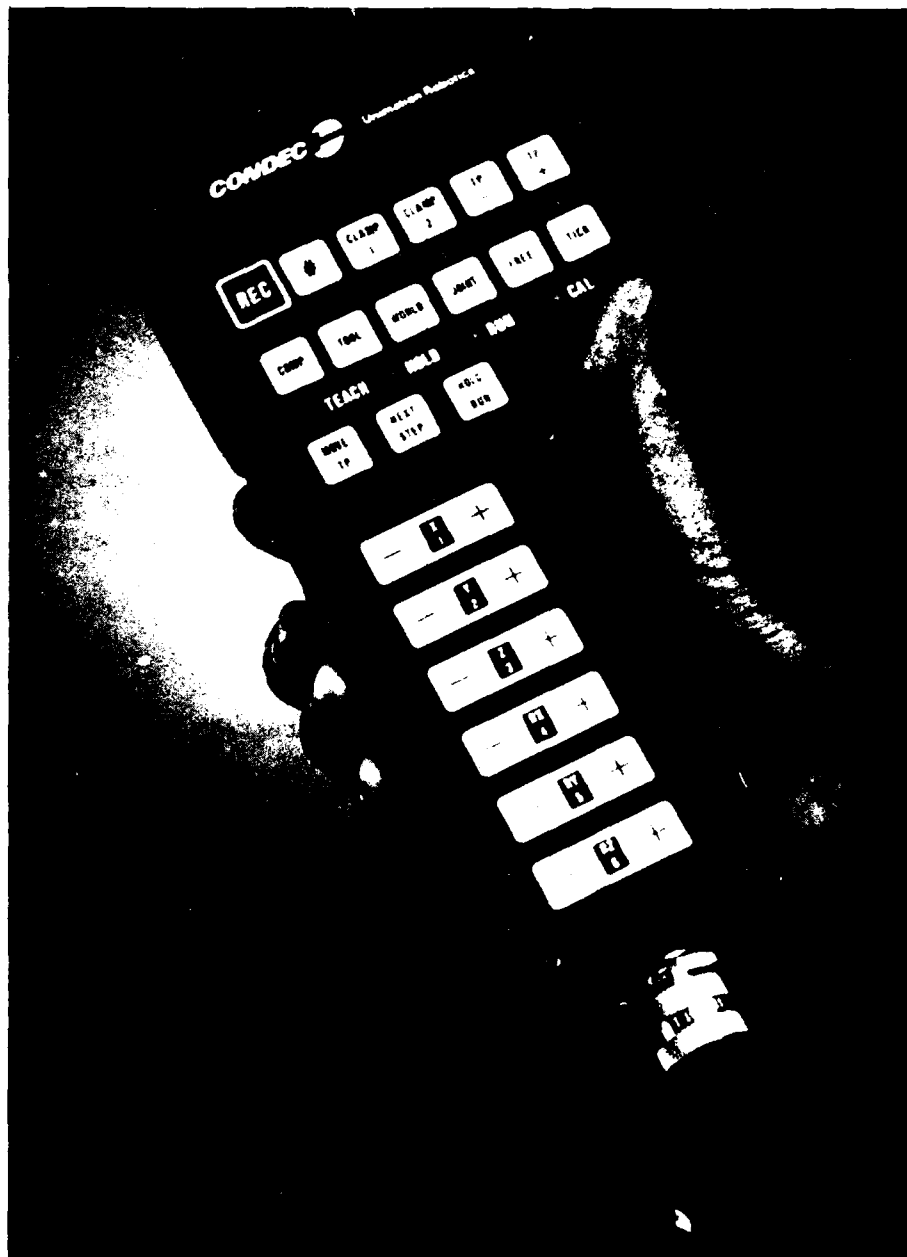


Figure 18. Unimation (Westinghouse) teach pendant.
(Courtesy of Westinghouse Electric Corporation)

Controls

A large black SAFETY HOLD push button on top of the pendant can be viewed as a "panic" stop button, though, when activated, power and data channels to the arm are still open. If pressed when the arm is in motion, the arm decelerates to a stop. On the pendant face there are 21 push buttons in three groups of 12, 3, and 6. On the pendant's left side is a spring-loaded Teach Speed switch labeled by an arrow and FAST. When it is in the middle, the arm moves at medium (normal) speed; when moved back toward the pendant holder, it moves at slow (creep) speed; when away from the pendant holder, the arm moves at high (jog) speed.

a. Upper group of 12:

Top row: (1) A red record push button (REC) inserts locations and/or motion instructions into the CMOS memory in the Teach mode (with appropriate VAL programming). (2) A second button is presently unused. (3) Two CLAMP buttons, 1 and 2, operate the end effectors, such as a gripper; one button alternately opens and closes it, and the other can activate some other action. (4) Two buttons labeled TP+ and TP- initiate the Move mode and step through taught points alphabetically in a positive or negative direction; the names of the points appear on the display. Then pressing the MOVE TP button can move the tool to the desired location when it appears.

Second row: This consists of six "preconditioning" buttons, each with a light that goes on when it is pressed. (1) A COMP button places the arm under computer rather than teach pendant control; the robot must be under computer control during startup and then the CRT/keyboard terminal is controlling the arm movement. (2) A TOOL mode button assigns the X, Y, and Z positioning buttons (located below) to move the tool mounting flange on the hand along the straight lines of the tool coordinate system axes, and the RX, RY, and RZ orienting buttons (also located below) to rotate the hand around the tool coordinate system axes. (3) A WORLD mode button assigns the X, Y, and Z positioning buttons to move the mounting flange and tool along straight lines parallel to world coordinates, and RX, RY, and RZ orienting buttons to rotate the flange and tool around lines parallel to those coordinates. (4) A JOINT mode button assigns buttons X, Y, Z, RX, RY, and RZ (also labeled 1, 2, 3, 4, 5, and 6) to move their respective joints individually (only) in either a positive or negative direction in rotation around the axis of the joint. (5) A FREE mode button frees five of the robot's six joints from servo control (not Joint 2), so they swing free (more or less); only one joint at a time can be placed in the Free mode. The robot segments can then be moved directly by hand. (6) A TICK mode button changes the speed of a motion in the Tool, World, or Joint modes, so the Teach Speed switch on the pendant's side at high speed moves the arm slightly slower than the lowest nontick speed and at medium speed slightly slower than this, while at low speed each press on the button makes the arm move only a single encoder bit rather than continuously.

b. Middle group of three in a row: (1) A MOVE TP button while pressed makes the robot's arm move in a straight line to the taught point showing on the pendant display at that moment. (2) A NEXT STEP button advances the program a single step each time the button is pressed. (3) A

HOLD RUN button halts moving the arm at the next program step and lights the HOLD indicator (see the section on Displays). Then pressing the button for 2 seconds until the RUN indicator lights makes the arm continue the program at the next step.

c. Lower group of six in a column on the pendant handle: These are the buttons labeled X, Y, Z, RX, RY, and RZ for coordinates and also numbered 1, 2, 3, 4, 5, and 6 for the joints mentioned earlier. They activate the arm or its joints. What they do depends on the mode that has been selected (preconditioned)--Tool, World, Joint, or Free (see the second row of the upper group of buttons on Figure 18). These are rocker switches, with - and + on each button, to indicate the direction of motion determined by a switch's left or right side.

Design Aspects

Coding/Grouping

The REC (record) button is coded red. The SAFETY HOLD button is not. The three sets of buttons with different functions are spatially separated, and the robot movement buttons are wider than the others and arranged in a column.

Labeling

Words longer than four letters are abbreviated. The speed switch on the side has no labeling. Plus and minus signs and their locations indicate movement direction on the robot movement buttons.

Feedback

Pressing any one of the six preconditioning buttons turns on an indicator light above it. Two indicator lamps above it come on when the HOLD RUN button is pressed, indicating which function has been activated. Feedback of mode selection is shown on the display. Each time the REC button is pressed to put a Move instruction in the program, the display blinks and the pendant beeps twice.

Errors

Hardware failures (not operator errors) are indicated on the display.

Multiple functions

Both CLAMP buttons open and close the gripper (or activate opposing actions in another end effector). The HOLD RUN button halts the robot and then with another press (lasting 2 seconds) starts it again. The six robot

motion buttons all impart one of two directions depending on which side is pressed, and have one of three functions depending on which mode button has been pressed.

Miscellaneous

The pendant shape is an inverted L; the lower portion is graspable by the left hand (or perhaps the right).

CRT/KEYBOARD TERMINAL

The terminal is a separate unit for the PUMA robots and the Unimate 2000 and 4000 and can rest on top of the controller cabinet; for the Unimate 6000 (an earlier version for PUMAs) it is recessed into the cabinet. Keys on the keyboard are Unimation's own industrial membrane type. There are a number of special keys color-coded in tan; standard keys are blue. With the Control key held down, eight letter keys can issue special commands: C aborts a current command; S stops output to the terminal to review it; Q resumes that output; O suspends that output; W slows that output; Z terminates program execution (unexpected end of file); R redisplay the current input line; and U cancels current input. The terminal is used to communicate through the VAL II language (see the section on Software). It functions as the Program Editor by which a user can create or modify programs (except that teaching movements to the robot must be done through the pendant).

CONTROLLER PANEL (Controller Front Panel and Controller Top Panel) (see Figure 19)

This unit has most of its controls and displays on its face, with some also on top (accessible when it is pulled out from the cabinet). (The controller panel for the Unimate 6000 has a different configuration.) Components are as follows:

- A POWER ON two-way toggle switch (1) sends power to the controller and lights a yellow lamp (2) above it.
- An ARM POWER ON push button (3) lights a yellow lamp (4) above it.
- A red ARM POWER OFF push button (for emergency stop) (5) turns off the ON light and locks the robot's brakes.
- A three-way RUN/HALT/RESTART selector switch (6) executes a program in the RUN position, stops it at the end of the current step in the HALT position, and resumes it at step one in the RESTART position.

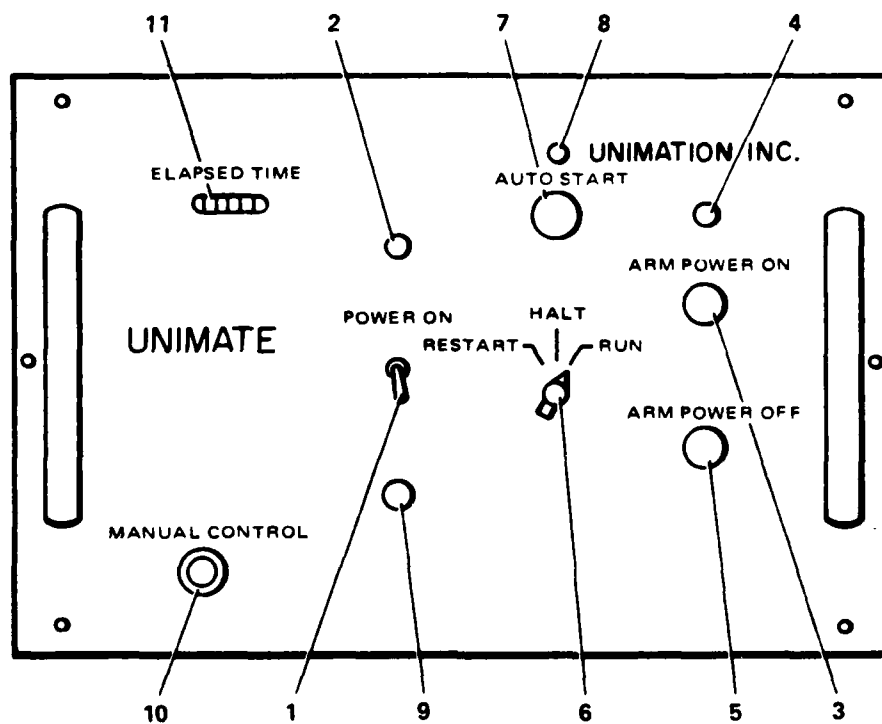


Figure 19. Unimation controller panel (legend omitted).
(Courtesy of Westinghouse Electric Corporation)

- An AUTO START push button (7) with a blue light (8), for system startup or command sequence to calibrate or run a program, eliminates having to start up via the terminal and a VAL program; it needs a user-written command program. There is also an ELAPSED TIME indicator (11).

On the top panel are (a) a Brake Release toggle switch to allow manual release of minor joint servomotor brakes; (b) a Run/Halt toggle switch to allow VAL to operate the Halt-Octal Debugging Tool, with a light for Run; (c) an INT toggle switch to reinitialize VAL; and (d) several amp and volt lights for joint errors.

SOFTWARE

VAL II, which followed VAL (there is also VAL Plus), is both a programming language and an operating system (monitor). VAL was "the first commercially available programming language" (Gruver et al., 1983) and was "an extension of the BASIC programming language." VAL II's changes included more extended sensory interfacing, formal network communication facilities, real-time path modification from internally and externally generated command signals, simultaneous robot and process control activities, combinations of joint and Cartesian-interpolated motions to produce long continuous motion, compound transformations, motion of the robot tool along a path algorithmically or mathematically described, and others. VAL had 65 monitor commands, 68 program instructions, and 82 (mostly error) messages. VAL II has 90 commands, 86 instructions, 152 messages (133 error, 5 warning, 14 informational), 40 real value functions, and 12 location functions. In VAL II, Version 2, the monitor can send messages to the teach pendant.

The VAL monitor gets commands from the CRT/keyboard terminal, the teach pendant, and user programs, and processes commands that control system status, define robot locations, store and retrieve information on a floppy disk, and create and edit robot control programs. It performs computations requested by user programs or from the pendant. It executes user programs and enables them to be created, edited, or stored.

VAL II commands define locations: Base (abbreviated BAS or B), Here (H), Point (PO), Teach (TE or T), Tool (TO), Mirror (MI), Touch Up Point (TUP), and Where (W) plus W1, W2, and W3. Fourteen are for program editing: Edit (ED), Change (C), Delete (D), Exit (E), Find (F), Insert (I), Last (L), Modify (M), Print (P), Replace (R1, R2), Skip To or Step (S), Teach (T), and Teach Straight Line (TS). Seven program and data listing commands are Directory (DIR), List1--location, Listp--program, Liste--expression, Llist, Plist, and Rlist. Fourteen program and data storage commands are Compress (COMP), Deletel, Fdelete (FD), Flist (FL), Format (FO), Load, Lstore, (LS), Pstore (PS), Restore (RS), Store, Store1, Storep, and Storer. (In an earlier listing for VAL, Load1 and Loadp appeared instead of some of these.) Copy and Rename are program manipulation commands. For program and data deletion, there are seven commands: Delete, Deletel, Deletel, Deleter, Ldelete (LD), Pdelete (PD), and Rdelete (RD). Seven program control commands are Abort (A), Do (D), Execute (EX), Proceed (PR), Retry (RET), Speed (SP), and Xstep (X); VAL also had commands (COMM)--under "command program control" in VAL II--and Next (N).

Seven system status and control commands are Calibrate (CA), Done (DON), Free (FR), Limp, PCstatus, Status (STA), and Zero (ZE--Z in VAL); VAL also had Bank and Diagnostic (DIA), while its Lower Limit (LL) and Upper Limit (UL) are under Miscellaneous in VAL II. There are 11 parameters and system switches: Continuous Path (CP), Disable (DIS), Disk.Net, Dry.Run Enable (EN), Interactive, Messages (ME), Parameter, Remote.Pin, and Switch (SW). VAL also had Cathode-Ray Tube Terminal (CRT) and Hand-Held Switch (HHT). VAL II has some additional commands for supervisory communications.

Fifteen VAL II program instructions for motion are Align, Appro, Appros, Brake (BRA), Break (BRE), Delay, Depart, Departs, Drive (DR), Move, Moves, Movest, Movet, Nest (NE), and Ready (READ). Three location-variable assignment instructions are Here (HE), Set (SE), and Tool (TO). Seven hand control instructions are Close, Closei, Grasp (GR), Open, Openi, Relax, and Relaxi; several are new. Eight program control instructions are Call (CAL), Goto (GO), Halt (HA), If, Pause (PAU), Return (RET), Stop (ST), and Wait (WA); VAL also had Gosub (GOS), If..Then, Ifs, Ignore (IG), React, Reacti, and Signal (SI). Seven trajectory control instructions are Coarse (CO), Fine (FI), intoff, Inton, Nonull (NON), Null (NU); and Speed (SP); VAL also had Break (BR). Six configuration control instructions are Above (AB), Below (BE), Flip (FL), Lefty (LE), Noflip (NOF), and Righty (RI). Other instructions occur in the categories of binary signals, structured constructs, asynchronous processing, input and output, path control communications, and miscellaneous: Attach (AT), Base (BA), Detach (DET), Disable (DI), Enable (ENA), Parameter (PAR), Pulse (PU), and Timer (TI).

Students learning VAL commands and instructions have been given a plastic card listing them all, with references to the pages in the User's Guide to VAL. Unimation is investigating the use of menus in place of sole reliance on remembering commands.

HUMAN FACTORS INTERESTS

According to M. J. Dunne (personal communication, August 1985), the pendant design has remained constant and there has been no human factors engineering study as such of pendant design; if there was to be a change, there would be inevitable problems in accustoming users to it, since Unimation robots have become so widely introduced. No consideration has been given to using a joystick instead of motion buttons; factory personnel are accustomed to buttons, and the device must be suited to their level of thinking in spatial relationships. Some feedback comes from users to trainers and designers. What might be viewed as a major human factors improvement ("user-friendly advance") in VAL was the development of a capability to move a robot in its entirety along Cartesian coordinates, instead of just joint by joint. This might be compared to the advance, in computer programming, from machine to assembly language, or from assembly language to higher order (compiler and interpreter) languages.

11. HITACHI, LTD., TOKYO, JAPAN

Detailed information has been obtained only from brochures describing the teach pendants for the A Series A4010H assembly robot and the A Series PTP type A3020P and A4020P. The teach pendant (teach box) for the A4010 is described first but no illustration of this pendant is available.

A4010H ROBOT TEACH PENDANT

The 4010H has four degrees of freedom, plus an end effector (hand). Arm 1 and Arm 2 rotate horizontally (around vertical axes), and a wrist that moves up and down and rolls (twists). This robot can be taught by direct manual manipulation (an operator moves it directly) or with a teach pendant, called a teach box, or both, with the pendant being used for fine adjustment after the robot is pushed or pulled by hand.

Displays

These consist of a power-on light, LEDs associated with buttons, and an LED display that shows a program number (two characters) and an instruction code (three characters).

Controls

In one array there are ten "Operator" keys with white letters unless otherwise noted: (a) an EMERGENCY STOP (with red label and outline); (b) a CONTROL key with LED; (c) a ZERO ADJ key with LED; (d) a DIRECT TEACH key with LED; (e) a POS ADJ key (with yellow label) with LED; (f) a PROG TEACH key (with blue label); (g) a PLAYBACK key; (h) a REPEAT key; (i) a STEP key; and (j) a STOP key (blue label).

In a second array there are 12 dual-use command keys with green labels (unless otherwise noted): (a) JMP (blue); (b) IF (blue); (c) OUT (blue); (d) INDXD/MOV (S); (e) P-NO/SET; (f) P-NO/INC; (g) MOV (F); (h) MOV (S); (i) INDXD/MOV (F); (j) HAND/TIMER; (k) MO; and (l) C (yellow). These also are numbered 0 through 9 for their other use, except for the last two. The MON has UP as its shared function.

In a third array, to the right of the second, there are five rocker keys, with yellow labels, for POS ADJ (moving the robot). Two move ARM 1 and ARM 2, left or right, with horizontal arrows and letters L and R designating direction. One moves the wrist (not specified) up or down with vertical arrows. A fourth rolls the wrist TWIST with circular arrows. The fifth, HAND, has ON and OFF in circles. An ENTER key is below the second array.

An operator first presses the DIRECT TEACH key and moves the robot by hand to a "working position." Then he presses the POS ADJ key and one of the keys in the third array. Next he presses the PROG TEACH key and after it a command key (e.g., MOV[F]). For playing back, he presses the PLAYBACK key and after that the REPEAT and STEP keys.

A3020P AND A4020P TEACH PENDANT (see Figure 20)

The A3020P robot and A4020P have similar designs, but their pendant (Dedicated Programming Unit) is more elaborate.

Displays

Four of the keys have associated LEDs. A screen can display 7 lines of 40 characters each. It supports a menu system for commands.

Controls

A large EMERGENCY STOP button is at the top right corner, and underneath it is an LED for CPU ERROR. Two scrolling buttons are on the left of the screen. Seven green soft function buttons are under the display, labeled F1 through F7. There are four other arrays of buttons:

- a. A numeric keypad (black labels on white)
- b. A column of four keys, with labels and borders in yellow (on black), MAN SPEED, COORD, HAND 1 ON, and HAND 2 ON. The first has three LEDs to its right labeled P, L, and H, presumably to select speed. The second has two LEDs labeled C'IAN and ART, presumably to select the robot motion between Cartesian and individual joint movements
- c. Two columns of four keys each, with the following labels (green with green borders unless otherwise noted): MODE CHNG, DISP, SGL, STEP, REPEAT, TRANS, TEACH (black on yellow), STOP (red on black), and ERROR RESET (red on black)
- d. A column of four rocker keys to move the robot (black on white with a yellow band in the middle). The top key, TW (in the yellow band), has two arrowed circles on the sides. The next key, with UP DN in the middle, has UP and DN on the sides. The third key has ARM 2 in the middle with FWD and BWD on the sides. The bottom key has ARM 1 in the middle with L and R on the sides

In addition, DEL and SET keys and two vertical arrow keys are black on white (function not established). The soft keys acquire functions shown above them on the screen. For example, one menu consists of MOVE, HAND, SPEED, WAIT, DELAY, and NEXT. The last can be pressed to display the remaining part of the menu level, if any.

Design Aspects

These two pendants have an abundance of color-coding, in addition to spatial separations between sets of buttons. Illustrations of the controller panels also indicate considerable color-coding of buttons.

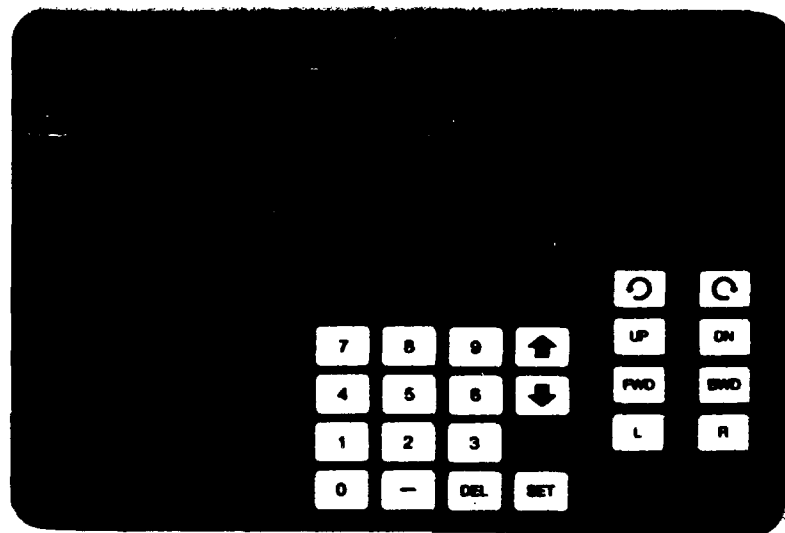


Figure 20. Hitachi teach pendant for A3020P and A4020P robots.
(Courtesy of Hitachi, Ltd.)